the required headcount. The required headcount is based on the total work to be performed, the expected productivity of performing that work, and the time to the phase deadline. Gross Labor applied to the project phase is the product of the headcount and workweek is the gross labor . Fractions calculated in the Labor Allocation sector are used to allocate the gross labor among basework, rework, quality assurance, and coordination.

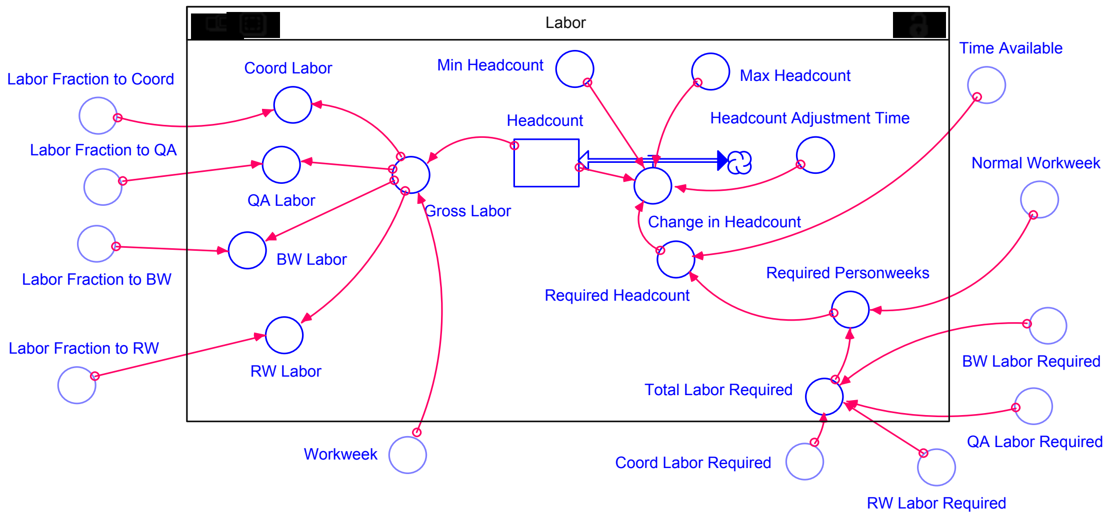

Figure 3-14: The Gross Labor Sector

The equations used to model the gross labor sector are described next.

Gross_Labor(Phase)=Headcount(Phase)*Workweek(Phase)

The gross labor available in a phase is the product of the headcount and the average workweek of the developers in the phase.

Headcount(Phase)=Headcount(Phase)+dt*(Change_Headcount(Phase))

Headcount(Phase)=Inital_Headcout(Phase)

Change_Headcount(Phase)=MIN(Max_Headcount(Phase)-Headcount(Phase),(Required_Headcount(Phase)-Headcount(Phase))/Headcount_Adjustment_Time(Phase))

A phase's headcount moves from the headcount at which the phase begins toward the headcount required by the phase. A maximum headcount for the phase limits the rate of headcount growth. The movement of the current headcount is smoothed and slowed by a time which represents the delay in adding developers to or releasing developers from a project phase.

```
Required_Headcount(Phase)=FIFZE((Required_Personweeks(Phase)*
Budget_Rqrd_Headct_effect)/(Time_to_Deadline(Phase)),0,
StoppedFlag(Phase))

Required_Personweeks(Phase)=Total_Labor_Required(Phase)/Normal_Workweek(Phase)

Budget_Rqrd_Headct_effect=TABHL(TC1,Budget_Status,-1.00,0,0.10)
T TC1=0.00/0.05/0.10/0.20/0.25/0.35/0.40/0.50/0.7/0.90/1.00
```

The required headcount is the number of developers required to complete the remaining work by the phase's deadline if all developers work a normal workweek (typically 40 hours per week). A project over budget will reduce this number in an attempt to control spending. This is based on the assumption that project managers will feel pressure to keep headcount and thereby costs down to bring project costs closer to the project budget.

```
Total_Labor_Required(Phase)=RemainingWork(Phase)/AvgPrody(Phase)

RemainingWork(Phase)=Task_List(Phase)-Tasks_Released(Phase)

AvgPrody(Phase)=(Ref_Coord_Prdctvty(Phase)+Ref_BW_Prdctvty(Phase)+Ref_RW_Prdctvty(Phas
e)+Ref_QA_Prdctvty(Phase))/4
```

The labor required to complete a phase is the sum of the unfinished tasks divided by the average productivity of the four development activities. The remaining work includes basework, rework and tasks waiting for quality control. The productivities are weighted equally on the assumption that developers are not likely to incorporate the amount of each kind of work remaining in their estimates of total labor required.

The Labor Allocation sector (Figure 3-15) calculates the fraction of the gross labor which developers apply to the basework, quality assurance, rework, and coordination activities. For each of these activities the work available from the process and the productivity expected by developers of each activity are used to determine a pressure for applying labor to each activity. These pressures are adjusted based on the priority of the activity and the performance relative to targets for schedule, quality, and cost. The labor fractions are the fraction of the total pressure for each activity.

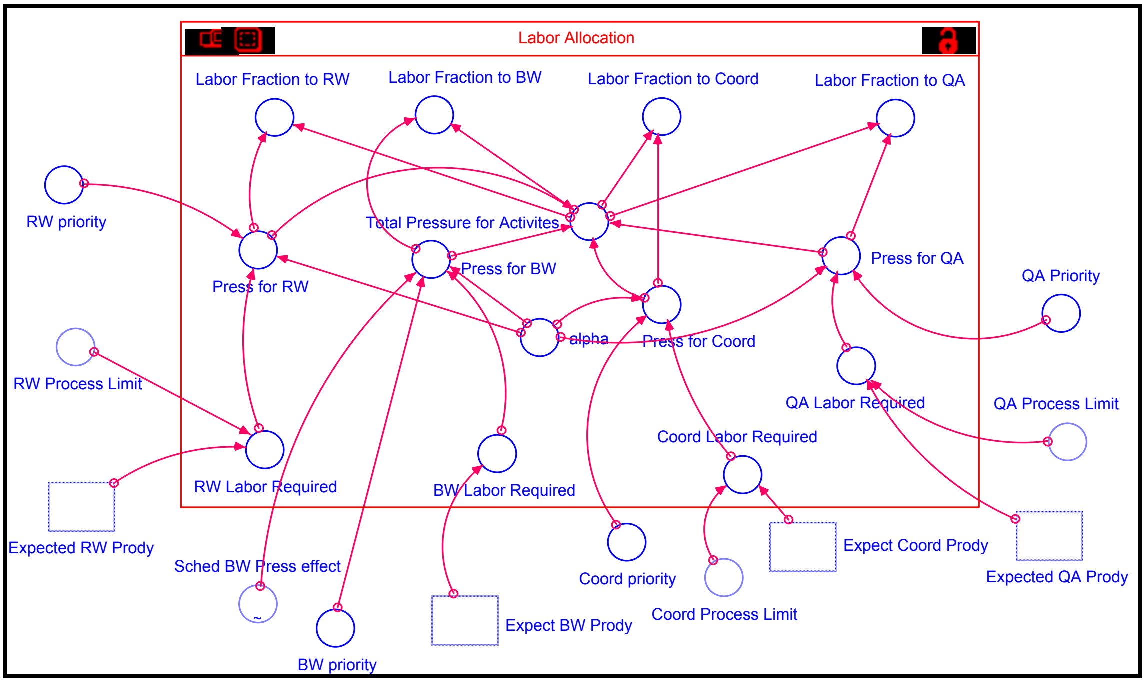

Figure 3-15: The Labor Allocation Sector

The equations used to model the allocation of labor are described below.

Coord_Labor_Required(Phase)=Coord_Process_Limit(Phase)/
Expect_Coord_Prdctvty(Phase)

BW_Labor_Required(Phase)=BW_Process_Limit(Phase)/Expect_BW_Prdctvty(Phase)

RW_Labor_Required(Phase)=RW_Process_Limit(Phase)/Expect_RW_Prdctvty(Phase)

QA_Labor_Required(Phase)=QA_Process_Limit(Phase)/Expected_QA_Prdctvty(Phase)

The labor required for each development activity is the number of person-hours required to complete the currently available work as determined by the process limit. This is the activity's process limit divided by the developer's expected productivity for that activity.

Press_for_BW(Phase)=EXP(((BW_Labor_Required(Phase)*BW_Priority(Phase)*
Sched_BW_Press_effect(Phase)*Cost_effect_on_BW_Import)/alpha(Phase)))

Press_for_RW(Phase)=Quality_Goal_Switch*(EXP((RW_Labor_Required(Phase)*
RW_Priority(Phase)*Qual_Gap_effect_on_QARW_priority)/alpha(Phase)))

Press_for_Coord(Phase)=Quality_Goal_Switch*((EXP((Coord_Labor_Required(Phase)*
Coord_Priority(Phase)*Qual_Gap_effect_on_Coord_Import)/alpha(Phase))))

Press_for_QA(Phase)=Quality_Goal_Switch*(EXP((QA_Labor_Required(Phase)*
QA_Priority(Phase)*Qual_Gap_effect_on_QARW_priority)/alpha(Phase)))

The pressure felt by the developers to use their available time for each of the development activities increases exponentially with increases in the labor required and the priority given to the activity. Low quality performance relative to the quality standard increases the pressure for rework, quality assurance and coordination.

Qual_Gap_effect_on_QARW_priority(Phase)=TABHL(TQ2,Current_Quality(Phase)-
Quality_Goal(Phase),-1.00,0.00,0.10)
T TQ2=2.20/2.14/2.07/1.99/1.90/1.80/1.68/1.54/1.38/1.20/1.00

Current_Quality(Phase)=1-(Known_Rework(Phase)/(Tasks_Released(Phase)))

The quality performance in each phase influences the priority given to quality assurance and rework in the allocation of labor. The current quality is the from the perspective of the developer and is the percent of tasks released which are believed to be free of errors. When the current quality is greater than the quality goal (described later) the priorities given to rework and quality assurance are not changed. But when current quality is worse than the quality goal the priority of rework and quality assurance increases to reflect the developer's attempts to correct for poor quality by putting more of the available time into the development activities which most directly influence the number of errors. An upper limit to this influence is assumed to exist and be a factor of 2.20.

Sched_BW_Press_effect(Phase)=TABHL(TL7,Sched_Pressure(Phase),0,5,0.50)
T TL7=1.00/1.01/1.04/1.07/1.12/1.17/1.25/1.32/1.44/1.58/1.73

Poor schedule performance is modeled with schedule pressure (described later). This table can reflect a specific development organization's perspective of schedule impacts. In this table no level of schedule pressure decreases the priority of basework, reflecting a strong emphasis on schedule performance. As the schedule pressure increases the priority given to basework increases. This represents the developer's efforts to finish the project more quickly by focusing their available labor on the most direct and obvious reflection of progress. This representation reflects a possible simplification of the project in the mental models of the developers in that it assumes that at an operational level developers perceive increased basework as the only change in labor priority appropriate when a project gets behind schedule.

Labor_Fraction_to_Coord(Phase)=Press_for_Coord(Phase)/
Total_Pressure_for_Activites(Phase)

Labor_Fraction_to_BW(Phase)=Press_for_BW(Phase)/
Total_Pressure_for_Activites(Phase)

Labor_Fraction_to_RW(Phase)=Press_for_RW(Phase)/
Total_Pressure_for_Activites(Phase)

Labor_Fraction_to_QA(Phase)=Press_for_QA(Phase)/
Total_Pressure_for_Activites(Phase)

Total_Pressure_for_Activites(Phase)=Press_for_QA(Phase)+
Press_for_Coord(Phase)+Press_for_BW(Phase)+Press_for_RW(Phase)

The fraction (as a percentage) of the total available labor allocated to each of the development activities is the fraction of the total labor pressure felt by developers to perform the specific development activity. The total pressure for labor is the sum of the pressures for the four development activities.

BW_Labor(Phase)=BW_Labor(Phase)+dt*(Labor_Fraction_to_BW(Phase)*
Gross_Labor(Phase)/BW_Labor_Delay(Phase)-BW_Labor(Phase))

RW_Labor(Phase)=RW_Labor(Phase)+dt*(Labor_Fraction_to_RW(Phase)*
Gross_Labor(Phase)/RW_Labor_Delay(Phase)-RW_Labor(Phase))

Coord_Labor(Phase)=Coord_Labor(Phase)+dt*(Labor_Fraction_to_Coord(Phase)
*Gross_Labor(Phase)/Coord_Labor_Delay(Phase)-Coord_Labor(Phase))

QA_Labor(Phase)=QA_Labor(Phase)+dt*(Labor_Fraction_to_QA(Phase)*
Gross_Labor(Phase)/QA_Labor_Delay(Phase)-QA_Labor(Phase))

The labor allocated to each development activity is the smoothed product of the activity's labor fraction and the gross labor available. The first order smooth represents the delay between the generation of the pressure for a development activity and the allocation of labor to that activity. This delay is an estimate of the effects of organizational inertia and the flexibility of developers to shift from one development activity to another.

Modeling the size of the average workweek and the effects of fatigue are done by the Workweek sector (Figure 3-16). The current workweek moves within a range (0 - 140 hours per week) relative to a normal workweek (40 hours per week) based on the impacts of schedule pressure. This value is averaged over the recent past to represent developer fatigue, which impacts the quality of work and productivity.

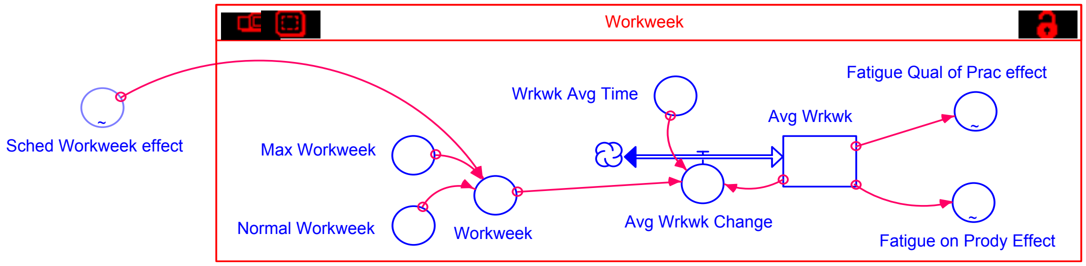

Figure 3-16: The Workweek Sector

The equations used to model the workweek sector are described next.

Avg_Wrkwk(Phase)=Avg_Wrkwk(Phase)+dt*(Avg_Wrkwk_Change(Phase))

Avg_Wrkwk_Change(Phase)=(Workweek(Phase)Avg_Wrkwk(Phase))/
Wrkwk_Avg_Time(Phase)

The average workweek in each phase moves from the initial normal workweek (40 hours per week) toward the current workweek at a rate slowed by the time required for working more or less than the average to have an impact on the developers. For example if the average workweek was 40 hours per week and the developers were relatively rested and the current workweek suddenly jumped to 60 hours per week the impacts would not be immediately felt by the developers. But as the current workweek remained high fatigue would develop slowly. The slow growth in the average workweek reflects this change.

Workweek(Phase)=MIN(Normal_Workweek(Phase)*
Sched_Workweek_effect(Phase),Max_Workweek(Phase))

Sched_Workweek_effect(Phase)=TABHL(TL10,Time_Required(Phase)/
Time_to_Deadline(Phase),0,5,0.5)
T TL10=0.97/0.99/1.0/1.05/1.15/1.30/1.50/1.80/2.20/2.70/3.30

The workweek is the developers response to the pressure to keep the project on schedule but remains within a maximum workweek size. There is very little response to being ahead of schedule, when the ratio of time required to the time available (the schedule pressure) is less than 1. This is consistent with the table reflecting the developer response to schedule pressure on basework priority. But as the schedule pressure increases the increase in workweek grows exponentially to a maximum multiplier of 3.30. This represents and extreme condition of developers feeling pressure to work 132 hours per week (at 40 hour per week normal workweek). While actually working this many hours is unreasonable, developers feeling the pressure to work this many hours in response to extremely high schedule pressure is possible and a reasonable upper bound.

3.3.4.2 The Experience Sector

The Experience sector (Figure 3-17) models the cumulative experience of the development team and its impact on productivity. Cumulative experience is measured in experience units which are the same size as development tasks. These are accumulated by the development team by performing basework and by increases in headcount, which represent the addition of more developers to the project phase. Accumulated experience decreases with decreases in headcount, representing developers leaving the project phase. The accumulated experience affects the productivity through an exponential learning curve relationship in which the productivity increases at a decreasing rate with more and more experience.

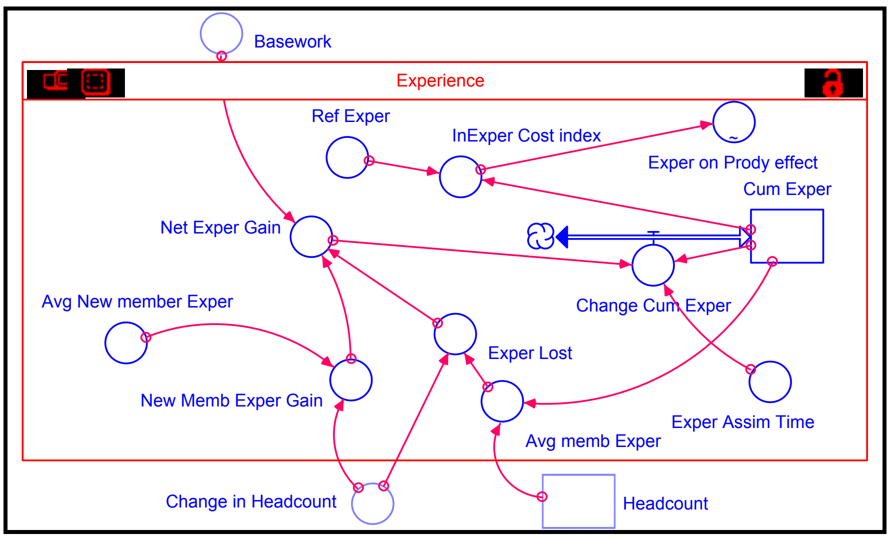

Figure 3-17: The Experience Sector

The equations used to model the experience sector are described next.

Cumm_Exper(Phase)=Cumm_Exper(Phase)+dt*(Change_Cum_Exper(Phase))

Cumm_Exper(Phase) [initial value]=Avg_New_member_Exper(Phase)*
Inital_Headcout(Phase)

Change_Cum_Exper(Phase)=Net_Exper_Gain(Phase)/Exper_Assim_Time(Phase)

Experience is measured in units which each represent the lesson learned from performing a single development task. The cumulative experience of the developers working on each phase at the beginning of the phase is assumed to be the product of the starting headcount of the phase and the experience level of a new development team member. The (effective) cumulative experience of the team adjusts to actual changes in the team experience at a rate slowed by the time required to assimilate those changes into its development work. This time represents the time to translate experience into useful knowledge and the time required to start to apply that useful knowledge.

Net_Exper_Gain(Phase)=Basework(Phase)+New_Memb_Exper_Gain(Phase)-Exper_Lost(Phase)

The actual experience of the team increases with each performance of a new development task. It is assumed that learning occurs primarily in the performance of initial development work and that no additional experience is gained by quality assurance, rework, or coordination. The actual experience of the team also increases with the additional of new team members and decreases with the loss of team members.

New_Memb_Exper_Gain(Phase)=Avg_New_member_Exper(Phase)*
MAX(0,Change_Headcount(Phase))

Exper_Lost(Phase)=(-1)*MIN(0,Change_Headcount(Phase))*Avg_memb_Exper(Phase)

Avg_memb_Exper(Phase)=FIFGE(Cumm_Exper(Phase)/
Headcount(Phase),Cumm_Exper(Phase),Headcount(Phase),1.0)

New team members add experience at a low "new member" level. Experience is lost at the rate of the average expereince level achieved by the development team.

Exper_index(Phase)=(0.80)**(LOGN(Cumm_Exper(Phase)/Ref_Exper(Phase))/LOGN(2))

Team experience influences other project features through a "learning curve" effect. The learning curve effect is modeled by calculating an influence factor which shrinks 20% for every doubling of cumulative experience of the team. The model uses a reference experience level and the team's cumulative experience to determine the number of doublings. The Experience Index decreases as the development team gains experience.

3.3.4.3 The Quality of Practice Sector

The Quality of Practice sector (Figure 3-18) models the impacts of schedule (working faster), experience (working smarter), coordination (working more effectively with others), and fatigue (working tired) on the quality of work performed by the developers. A reference level of quality is adjusted up with more coordination and experience and down with more schedule pressure and fatigue. A phase's quality of practice impacts its error generation and discovery rates.

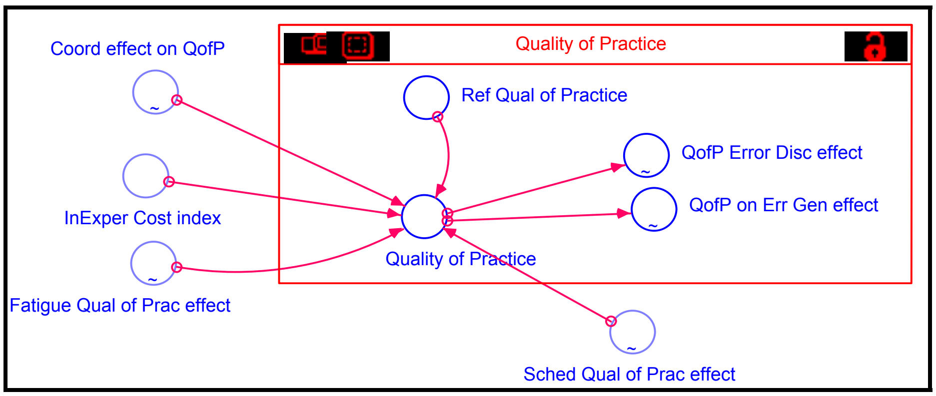

Figure 3-18: The Quality of Practice Sector

The equations used to model the quality of practice sector are described next.

Quality_of_Practice(Phase)=Ref_Qual_of_Practice(Phase)*Exper_effect_on_QofP(Phase)*
Sched_Qual_of_Prac_effect(Phase)*Fatigue_Qual_of_Prac_effect(Phase)*
Coord_effect_on_QofP(Phase)

The quality of practice of the developers in each phase is based on a reference value which is altered by developer experience, schedule pressure, fatigue, and coordination. Each of the influences represents a different cause of changes in the quality of practice. More experience improves the quality of practice because developers are "working smarter".

Exper_effect_on_QofP(Phase)=TABHL(T7,Exper_index(Phase),0,5,0.50)
T T7=2.50/2.4/2.2/1.9/1.5/1.00/.8/.6/5/.4/.35

More experience improves the quality of practice. This relationship reflects that developers are "working smarter" as a team within a phase when they have more experience. Early in the project when the development team has relatively little experience and the index is large the lack of experience decreases the team's quality of practice. As the team gains experience by performing basework the index drops and the effect on the quality of practice improves to a maximum impact of increasing the quality of practice by a factor of 2.50.

```
Sched_Qual_of_Prac_effect(Phase)=TABHL(T11,Sched_Pressure(Phase),1,10,0.90)
T T11=1/0.99/0.97/0.94/0.90/0.85/0.79/0.72/0.64/0.55/0.45
```

Increasing schedule pressure decreases the quality of practice because the developers are "working faster" to recover the time and therefore not doing as good a job. Schedule pressure is assumed to only hurt and never help the team's quality of practice. A lower limit is placed on this relationship, reflecting the assumption that professional developers will retain some quality of practice even under extremely high schedule pressure conditions.

```
Fatigue_Qual_of_Prac_effect(Phase)=TABHL(TL12,Avg_Wrkwk(Phase)/
Normal_Workweek(Phase),0,5,0.50)
T TL12=1.05/1.05/1.0/0.98/0.94/0.88/0.80/0.70/0.58/0.44/0.44
```

More fatigue decreases the quality of practice because the developers are "working tired". Fatigue is modeled as the response to the ratio of the average workweek over a period of time (described above) to the normal workweek. The relationship of fatigue to the quality of practice is nonlinear with little influence when the average workweek is less than normal. This reflects the assumption that any time made available due to needing to work on the project less than the normal workweek will be absorbed (according to Parkinson's Law). A maximum effect is reached as the average workweek exceeds the normal by a factor of five.

```
Coord_effect_on_QofP(Phase)=TABHL(TL3,Coord_Status(Phase),0,2,0.20)
T TL3=0.00/0.06/0.18/0.36/0.6/0.9/1.28/1.66/1.84/1.96/2.00
```

More coordination increases the quality of practice because the developers are "working more effectively". This is because their interaction with other development phases has given them improved knowledge and insight into their part of the entire project. Coordination only improves the quality of practice in this relationship.

```
QA_Status(Phase)=QA_Labor(Phase)/(QA_Labor_Required(Phase))
```

The status of the quality of practice in each of the phases is the ratio of the labor provided for quality assurance to the labor required for quality assurance work.

3.3.4.4 The Expected Productivity Sectors

The four Expected Productivity sectors (Figures 3-19, 3-20, 3-21, and 3-22) model developer's perceptions of the productivity of development activities. These expectations are used to simulate the pressures felt by developers to apply their time to the different development activities. Each of the four perceived productivities (for basework, quality assurance, rework, and coordination) are based on the actual productivity experienced and altered by the delay in reporting and adjusting of productivity expectations.

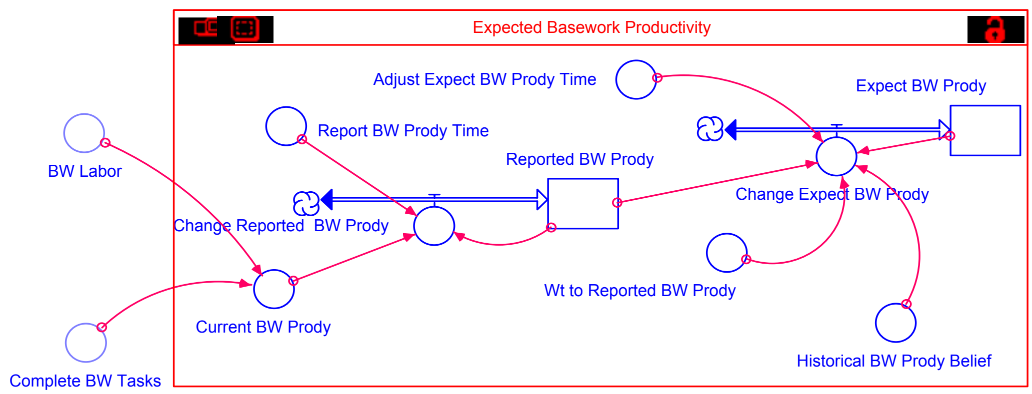

Figure 3-19: The Expected Basework Productivity Sector

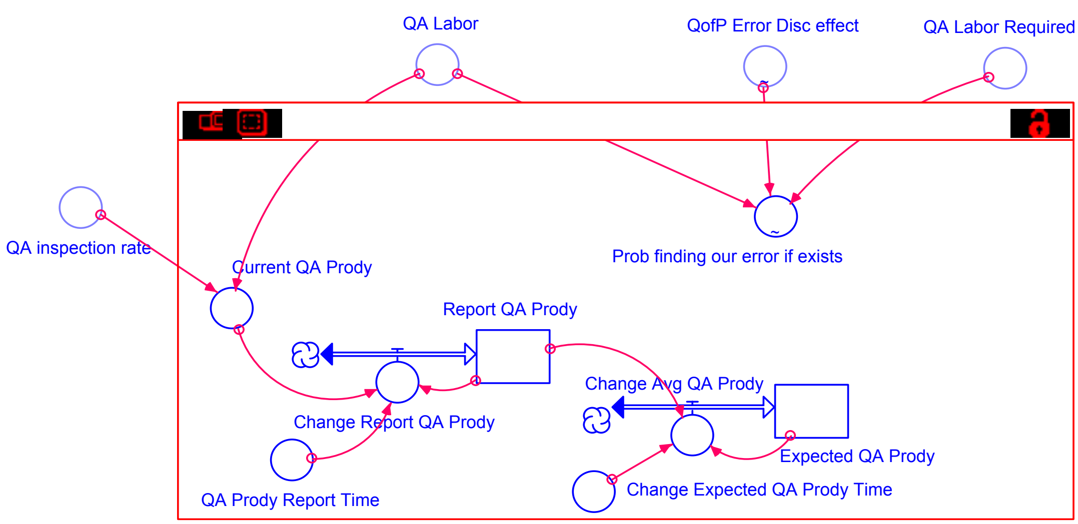

Figure 3-20: The Expected Quality Assurance Productivity Sector

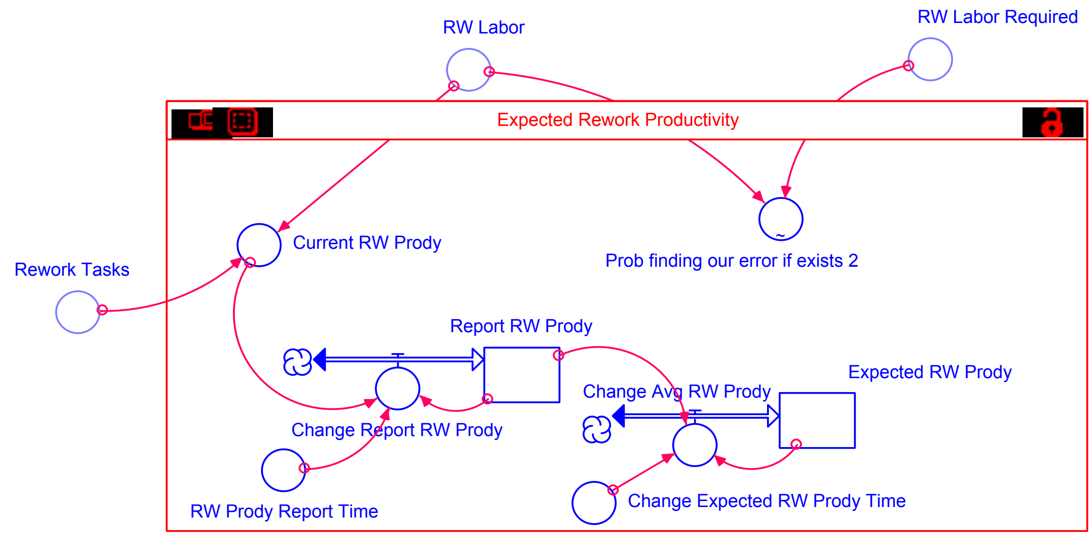

Figure 3-21: The Expected Rework Productivity Sector

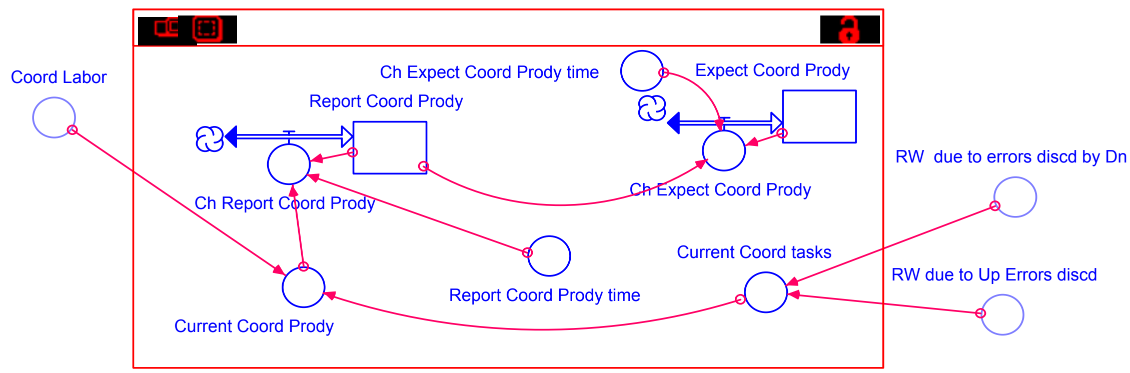

Figure 3-22: The Expected Coordination Productivity Sector

The equations used to model the expected productivity sectors are described next.

Expect_BW_Prdctvty(Phase)=Expect_BW_Prdctvty(Phase)+dt*
(Change_Expect_BW_Prdctvty(Phase))

Expect_RW_Prdctvty(Phase)=Expect_RW_Prdctvty(Phase)+dt*
(Change_Expect_RW_Prdctvty(Phase))

Expected_QA_Prdctvty(Phase)=Expected_QA_Prdctvty(Phase)+dt*
(Change_Expect_QA_Prdctvty(Phase))

Expect_Coord_Prdctvty(Phase)=Expect_Coord_Prdctvty(Phase)+dt*
(Change_Expect_Coord_Prdctvty(Phase))

The developers in each phase generate expectations about their productivity at performing the four development activities. These expected productivites move from initial reference values which are estimates of historical productivity expectations toward actual productivites.

Change_Expect_BW_Prdctvty(Phase)=(Actual_BW_Prdctvty(Phase)-Expect_BW_Prdctvty(Phase))/Adjust_Expect_BW_Prdctvty_Time(Phase)

Change_Expect_RW_Prdctvty(Phase)=(Actual_RW_Prdctvty(Phase)-Expect_RW_Prdctvty(Phase))/Adjust_Expect_RW_Prdctvty_Time(Phase)

Change_Expect_QA_Prdctvty(Phase)=(Actual_QA_Prdctvty(Phase)-Expected_QA_Prdctvty(Phase))/Adjust_Expect_QA_Prdctvty_Time(Phase)

Change_Expect_Coord_Prdctvty(Phase)=(Actual_Coord_Prdctvty(Phase)-Expect_Coord_Prdctvty(Phase))/Adjust_Expect_Coord_Prdctvty_time(Phase)

Changes in productivity expectations are slowed by a time which represents the time needed for developers to build the newly experienced productivity of the current project into their expectations about how they will perform in the future. This can require the alteration of long-held and personally important performance images which the developers hold.

BW_Labor_Limit(Phase)=Act_BW_Prd(Phase)*BW_Labor(Phase)

RW_Labor_Limit(Phase)=Act_RW_Prd(Phase)*RW_Labor(Phase)

QA_Labor_Limit(Phase)=Act_QA_Prd(Phase)*QA_Labor(Phase)

Coord_Labor_Limit(Phase)=Act_Coord_Prd(Phase)*Coord_Labor(Phase)

The limit placed on the development activity rate for each of the four development activities in each phase is the product of the actual productivity of that activity in the phase and the amount of labor allocated to the activity. This assumes that management will not add more developers to a phase than the current available workload and perceived productivities indicate are needed. This assumption may be unrealistic in some circumstances. The model could be altered to accommodate overloading of personnel by adding a structure to model the increase in labor provided due to managerial decisions to overload a development phase. The actual productivities of each development activity in each phase are formulated similar to the expected productivites. They start at initial reference values which are estimates of historical productivites experienced by the developers. The productivities move toward the most current productivity at a slowed rate and influenced by experience and coordination (see previous description).

Current_BW_Prdctvty(Phase)=Basework(Phase)/(BW_Labor(Phase))

Current_RW_Prdctvty(Phase)=Rework(Phase)/(RW_Labor(Phase))

Current_QA_Prdctvty(Phase)=QA_inspection_rate(Phase)/(QA_Labor(Phase))

Current_Coord_Prdctvty(Phase)=Coord_Limit(Phase)/(Coord_Labor(Phase))

The current productivity of each development activity in each phase is the rate at which that activity is being performed divided by the amount of labor applied to the activity. If the activity rate is being limited by the development process structure the current productivity can be much lower that if the labor structures limit activity.

Change_Actual_BW_Prdctvty(Phase)=((Current_BW_Prdctvty(Phase))/
Avg_Act_BW_Prdctvty_Time(Phase))*Exper_on_Prdctvty_effect(Phase)*
Coord_effect_on_Prdy(Phase)

Change_Actual_RW_Prdctvty(Phase)=((Current_RW_Prdctvty(Phase))/
Avg_Act_RW_Prdctvty_Time(Phase))*Exper_on_Prdctvty_effect(Phase)*
Coord_effect_on_Prdy(Phase)

Change_Actual_QA_Prdctvty(Phase)=((Current_QA_Prdctvty(Phase))/
Avg_Act_QA_Prdctvty_Time(Phase))*Exper_on_Prdctvty_effect(Phase)*
Coord_effect_on_Prdy(Phase)

Change_Actual_Coord_Prdctvty(Phase)=((Current_Coord_Prdctvty(Phase))/
Avg_Act_Coord_Prdctvty_Time(Phase))*Exper_on_Prdctvty_effect(Phase)

Actual productivities move at a smoothed rate which represents the time required for instantaneous productivites to become effective (no smooth causes unrealistic large fluctuations) as effected by the level of team experience and coordination.

Exper_on_Prdctvty_effect(Phase)=TABHL(TL13,Exper_index(Phase),0,5,0.50)
T TL13=1.33/1.30/1.24/1.18/1.1/1.00/.9/0.82/0.76/0.72/0.70

More experience increases productivity. This relationship reflects that developers are "working smarter" as a team within a phase when they have more experience. The impact described by this relationship remains within a relatively narrow range (33% increase to 30% decrease in productivity). The lower limit reflects that productivity levels have minimums based on levels of training and skill which little experience cannot erode. The upper limit reflects a maximum impact of experience.

Coord_effect_on_Prdy(Phase)=TABHL(TL2,Coord_Status(Phase),0,1,0.10)
T TL2=0.195/0.41/0.575/0.725/0.825/0.89/0.945/0.96/0.975/0.985/1.00

The level of coordination also impacts productivity. Inadequate coordination decreases productivity because developers who do not communicate and coordinate their work are "working less effectively". This is because interaction with other development phases can give them improved knowledge and insight into their part of the entire project. Excess coordination is assumed to not impact productivity. Inadequate coordination has a limited reduction in productivity, reflecting that even developers who do not coordinate their work with other phases at all are able to proceed with development activities.

3.3.5 The Targets and Performance Subsystems

Like the Resources subsystems, the structural components of the Targets and Performance subsystems are based upon existing system dynamics models which are referenced in Table 3-1. The targets and performance subsystems model project goals and actual performance in three dimensions: time, quality and money. These subsystems consist of the Schedule, Quality, and Cost sectors.

3.3.5.1 The Schedule Sector

Schedule goals and performance are modeled at both the project and phase level. At the project level the time objective is set with the project deadline. Project performance in the time domain is measured by the actual project completion time. The Actual completion time is compared to the project deadline to evaluate project schedule performance.

The project deadline moves toward the projected completion date when the schedule pressure exceeds the capacity of the team to resist that pressure (Figure 3-23). Resistance to schedule slippage slows this movement. Schedule pressure is defined as the ratio of the time to the expected completion date to the time to the current project deadline. The expected completion date for the project is the expected completion date of the last development phase.

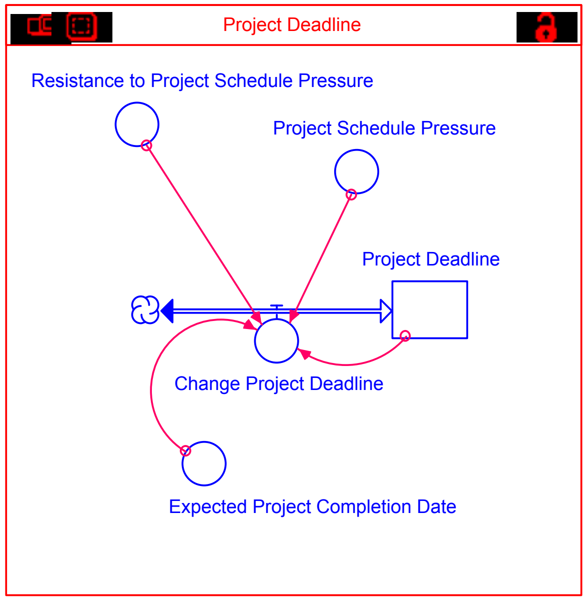

Figure 3-23: The Project Deadline Sector

A similar structure is used to model the schedule in each of the development phases. Expected phase completion dates are based on the remaining labor required and the available labor. Schedule pressure in each phase is defined the same as at the project level and impacts the allocation of labor within the phase, quality of practice, error discovery, and length of workweek.

The equations used to model the project and phase schedules are described next.

Project_Deadline=Project_Deadline+dt*(Proj_Sched_Slip)

Proj_Sched_Slip=FIFGE(Expected_Proj_Completion_Time-Project_Deadline,0,Proj_Sched_Press,Resistance_to_Sched_Slip)

The project deadline moves from its current value to the expected completion time when the schedule pressure for the entire project exceeds the threshold for schedule pressure acceptable.

Proj_Sched_Press=MAX((Expected_Proj_Completion_Time-TIME)/(Project_Deadline-TIME),(Project_Deadline-TIME)/(Expected_Proj_Completion_Time-TIME))

The schedule pressure is defined as the ratio of the time required to complete the project and the time available until the current project deadline. The maximum of the inverse ratios is used to capture schedule pressures due to expected completions before and after the current deadline.

Expected_Proj_Completion_Time=MAX(ExpComplTime(1),ExpComplTime(2),ExpComplTime(3),ExpComplTime(4),ExpComplTime(5))

The expected project completion time is the latest expected completion time of the development phases.

Project schedules of each of the development phases is modeled in the same manner as for the project as a whole.

DL(Phase)=DL(Phase)+dt*Change_DL(Phase)

Change_DL(Phase)=FIFGE(ExpComplTime(Phase)-DL(Phase),0,Sched_Pressure(Phase),Resistance_to_Sched_Slip)

The deadline for each phase moves from its current value to the expected completion time when the schedule pressure for the phase exceeds the threshold for schedule pressure acceptable.

Sched_Pressure(Phase)=MAX((ExpComplTime(Phase)-TIME)/(DL(Phase)-TIME),(DL(Phase)-TIME)/(ExpComplTime(Phase)-TIME))

The schedule pressure is defined as the ratio of the time required to complete the phase and the time available until the current phase deadline. The maximum of the inverse ratios is used to capture schedule pressures due to expected completions before and after the current deadline.

ExpComplTime(Phase)=ExpStartTime(Phase)+ExpDur(Phase)

The expected completion time of each phase is the time the phase is expected to start of actually starts plus the expected duration of the phase.

ExpStartTime(2)=((1-StartFlag(2))*((Task_List(1)/TotalTaskList)*Project_Deadline))+
(StartFlag(2)*Start_Time(2))

The time of the expected start time of each phase from the project start is estimated to be proportional to the size of phase to the project size if the phase has not started. This estimate is required because the exact starting time of phases are not known until the phase actually begins. The time of the expected start time of each phase from the project start for active or completed phases is the phase's actual starting time. This formulation must estimate the concurrence of dependent development phases. It cannot identify the longest dependent path of phases through the project and add expected durations along that path because that assumes totally sequential phases, which is a very false assumption and greatly exaggerates the expected completion time. The "StartFlag" parameter acts a switch between the estimate of future phases and the documentation of the start time of active or completed phases. Phase 2 is shown as an example of this formulation. Software limitations prevent the generalization of this formulation.

ExpDur(Phase)=Time_spent_to_Date(Phase)+Time_Required(Phase)

Time_Required(Phase)=RemainingWork(Phase)/(AvgPrody(Phase)*
Normal_Workweek(Phase)*(Max_Headcount(Phase)/2))

AvgPrody(Phase)=(Ref_Coord_Prdctvty(Phase)+Ref_BW_Prdctvty(Phase)+
Ref_RW_Prdctvty(Phase)+Ref_QA_Prdctvty(Phase))/4

RemainingWork(Phase)=MAX(0,(.999*Task_List(Phase))-Tasks_Released(Phase))

The expected duration of a phase is the time spent so far on the phase plus the estimate of the time required to complete the phase. The time spent to date is the time elapsed from the start of the phase. The estimate of the time required to complete the phase is the time necessary to complete the remaining work if the average historical (reference) productivities are applied by the average labor force. The average labor force is assumed to be the half the maximum headcount working the normal workweek. This formulation must differ from many in the model which reflect the developer and manager's perception of primarily the currently available development work. Developers and managers acknowledge the total remaining visible work in estimating required durations and see beyond the currently available work. Therefore this estimate must utilize the total work remaining to be done in the phase. The formulation assumes optimistic developers in that the forecast assumes that none of the unstarted basework will require rework.

3.3.5.2 The Project Quality Sector

The Project Quality sector (Figure 3-24) models the movement of the project quality goal from its initial level toward the current known quality level. Quality is measured in the percent of tasks which are known by the developers to be flawed. The time to adjust the quality goal slows the migration of the goal. Project quality below the project goal increases pressure for quality assurance, rework, and coordination work.

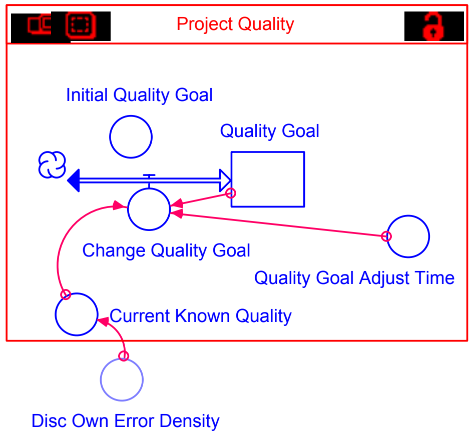

Figure 3-24: The Project Quality Sector

The equations used to model the project quality sector are described next.

Quality_Goal(Phase)=Quality_Goal(Phase)+dt*(Change_Quality_Goal(Phase))

Change_Quality_Goal(Phase)=(Current_Quality(Phase) - Quality_Goal(Phase))/Quality_Goal_Adjust_Time(Phase)

The quality goal of each phase moves from its initial value toward the current quality of that phase at a rate determined by the time required to adjust that goal. These movements are relatively small since the difference between the goal and actual quality is often relatively small and the adjustment time is relatively long.

Current_Quality(Phase)=1-(Known_Rework(Phase)/(Tasks_Released(Phase)+ Tasks_Completed_not_Checked(Phase)+Known_Rework(Phase)))

The current quality of each phase is the percent of tasks completed once which are believed to not be flawed. This reflects the developer's perspective (optimism) that unchecked tasks are not flawed. This formulation uses the complete project work instead of the release of tasks by the final project phase because during the project the latter information is not available to developers or managers.

Proj_Quality_Goal=Proj_Quality_Goal+dt*(Change_Proj_Quality_Goal)

Change_Proj_Quality_Goal=(Current_Proj_Quality-
Proj_Quality_Goal)/Proj_Quality_Goal_Adjust_Time

The quality goal of the entire project also moves from its initial value toward the current quality of the project at a rate determined by the time required to adjust that goal. These movements are relatively small since the difference between the goal and actual quality is often relatively small and the adjustment time is relatively long.

Current_Proj_Quality=1-(SUM(Known_Rework)/(SUM(Tasks_Released)+
SUM(Tasks_Completed_not_Checked)+SUM(Known_Rework))

The current quality of the project is the percent of tasks completed once which are believed to not be flawed. This reflects the developers perspective (optimism) that unchecked tasks are not flawed. This formulation uses the complete project work instead of the release of tasks by the final project phase because during the project the latter information is not available to developers or managers.

Proj_Quality_Gap=Current_Proj_Quality-Proj_Quality_Goal

The project's quality gap is the difference between the project quality goal and the current project quality.

Qual_Gap_effect_on_Coord_Import=TABHL(TQ1,Proj_Quality_Gap,-1.00,0.00,0.10)
T TQ1=2.10/1.90/1.72/1.56/1.42/1.30/1.20/1.12/1.06/1.02/1.00

The project quality status impacts the priority given by developers to the allocation of available labor to the coordination activity. Quality that exceeds the quality goal has no impact on coordination priority. But poor quality performance increases the priority of coordination in an "S" shaped curve with an upper limit of a multiplier of 2.10.

3.3.5.3 The Cost Sector

The Cost sector accumulates project costs, projects total project costs, compares them to the project budget, and influences several parameters depending on the difference. The equations used to model costs are described next.

Project_Cost_to_Date=SUM(Phase_Cost_to_Date)

Phase_Cost_to_Date(Phase)=Phase_Cost_to_Date(Phase)+dt*Straight_Cost(Phase)

The current cumulative cost of the project is the sum of the current cumulative costs of the project's phases. Each project phase cost rises with the addition of straight (salaried, no overtime premium) labor cost. Only straight time cost is used because the developers are salaried and are not compensated for overtime.

Straight_Cost(Phase)=Avg_Straight_Pay(Phase)*Straight_Time(Phase)*
Cost_Markup(Phase)

The incremental cost of straight labor is the product of the average hourly pay rate in dollars per hour, the time expended in hours, and the cost markup factor which represents the cost of overhead, fringe benefits, administrative support, employer-paid taxes, equipment, etc.

Straight_Time(Phase)=Headcount(Phase)*Normal_Workweek(Phase)

The amount of straight time is the number of developers currently working on the development phase times the hours per week used as the basis for payment (40 hours per week).

Tot_Exp_Costs=Forecasted_Costs+Project_Cost_to_Date

The total expected project cost is the sum of the cost of the project so far and the forecast of the remaining cost.

Forecasted_Costs=Avg_Cost*(Project_Deadline-TIME)

Avg_Cost=Project_Cost_to_Date/(TIME)

The forecasted cost is the average cost of the project to date extended to the current project deadline. This tends to increase the expected project cost as the project deadline slips.

Budget_Surplus=Proj_Budget-Tot_Exp_Costs

Budget_Status=Budget_Surplus/Proj_Budget

The budget surplus or deficit is the difference between the project's budget and the total expected cost of the project. The cost status of the project is represented in a form commonly used in practice, as the percent over (positive budget status) or under (negative budget status) budget.

Budget overruns impact other project factors. But being under budget has no impact. This is based on the assumption that developers do not believe that the underrun will exist for the duration of the project or that developers will use available funds by the end of the project. In either case, the underrun do not significantly influence the project.

```
Cost_effect_on_BW_Import=TABHL(TC2,Budget_Status,-1.0,0.0,0.10)
T TC2=1.87/1.58/1.35/1.17/1.06/1.00/0.98/0.95/0.89/0.81/0.65
```

Cost performance influences the priority of basework activity. This is based on the assumption that developers will try to finish a project more quickly to constrain costs. Therefore this relationship increases the priority of basework when projected costs exceed the budget.

3.3.6 Basis for Key Model Structures

The important model structures are based on previous system dynamics models, other project models and data based on field observations (described in more detail in Chapter 5). These are summarized in Table 3-1 below:

Model Structure	**Model References**	**Data based on Field Observations *(Yes/No)***
Process Structure and Scope Subsystems		
Development Tasks sector		
Demand-driven process limits	none	Yes
Exponential smooth of demand for activities	Hannon and Ruth, 1994	No
Internal Available-work constraint	Homer et al., 1993	Yes
External Available-work constraint	Homer et al., 1993	Yes
Recycling of flawed work	Cooper, 1980 Kim, 1988 Seville and Kim, 1993 Ford et al, 1993	Yes
Closed loop flow of tasks	Cooper, 1980 Richardson and Pugh, 1981 Ford et al., 1993	Yes
Interaction of phases	Cooper, 1980 Reichelt, 1990 Homer et al., 1993 Seville and Kim, 1993	Yes

Internal Errors sector		
Discovered vs. undiscovered errors	Cooper, 1980, 1993, 1994 Jessen, 1988, 1990	Yes

Table 3-1: Basis for Important Model Structure Components (partial)

Model Structure	Model References	Data based on Field Observations *(Yes/No)*
Resources Subsystem		
Gross Labor sector		
Perceived Resource needs	Jessen, 1988, 1990 Abdel-Hamid, 1984 Richardson and Pugh, 1981	Yes
Labor Allocation sector		
Priority weighted pressure	Abdel-Hamid, 1984	Yes
Workweek sector		
Average and side impacts	Kim, 1988	Yes
Fatigue effects	Homer, 1985 Abdel-Hamid, 1984	Yes
Experience sector		
Learning curve/productivity effect	Abdel-Hamid and Madnick, 1991	Yes
Limits and Productivity sectors		
Actual vs. perceived productivity	Richardson and Pugh, 1981 Jessen, 1988 Abdel-Hamid, 1984	Yes
Targets and Performance Subsystems		
Schedule sector		
Schedule pressure	Roberts, 1974	Yes
Perceived vs. actual progress	Roberts, 1974 Richardson and Pugh, 1981	Yes
Schedule estimates	Abdel-Hamid, 1984	Yes
Deadline slippage	Richardson and Pugh, 1981	Yes
Project Quality sector		
Quality goal slippage	Fiddaman, Oliva, and Aranda, 1993	No
Project Cost sector		
Cost estimates	Abdel-Hamid and Madnick, 1991	Yes

Table 3-1: Basis for Important Model Structure Components (continued)

3.4 Model Behavior

3.4.1 Introduction

Plots of simulation results over time can help develop an understanding of the model behavior. A few such results for two model configurations are provided to develop a fundamental understanding of the model. Exact parameter values and simulation output are not as important as an intuitive understanding of the impacts of the model's structure on its behavior. Simulation under specific sets of parameter values are addressed in the sensitivity analysis section and Chapter 5.

3.4.2 Typical Single Phase Model Behavior

Simulations of model behavior with a single phase and only the product development process and scope subsystems engaged provides a fundamental understanding of the system. Two plots will be shown. Figure 3-25 and 3-26 show the development tasks sector flows and stocks of this model configuration with no errors. The maximum basework rate set by the availability of tasks and the minimum duration is evident in Figure 3-25, as is the delay and smoothing impact of the structure on the Release Tasks flow. The resulting increase and then decline in the Completed not Checked stock and growth in the Tasks Released stock is seen in Figure 3-26.

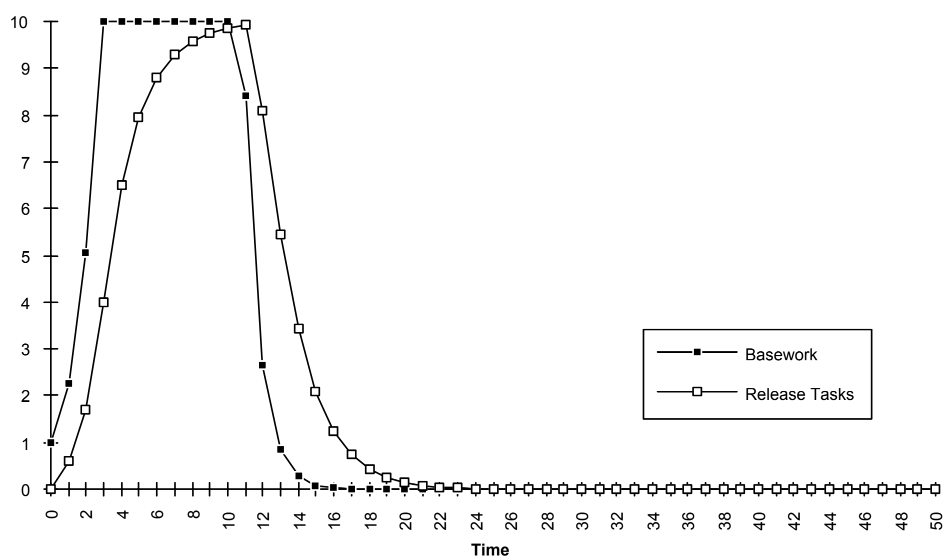

Figure 3-25: Single Phase Simulation with No Errors Development Task Sector Flows

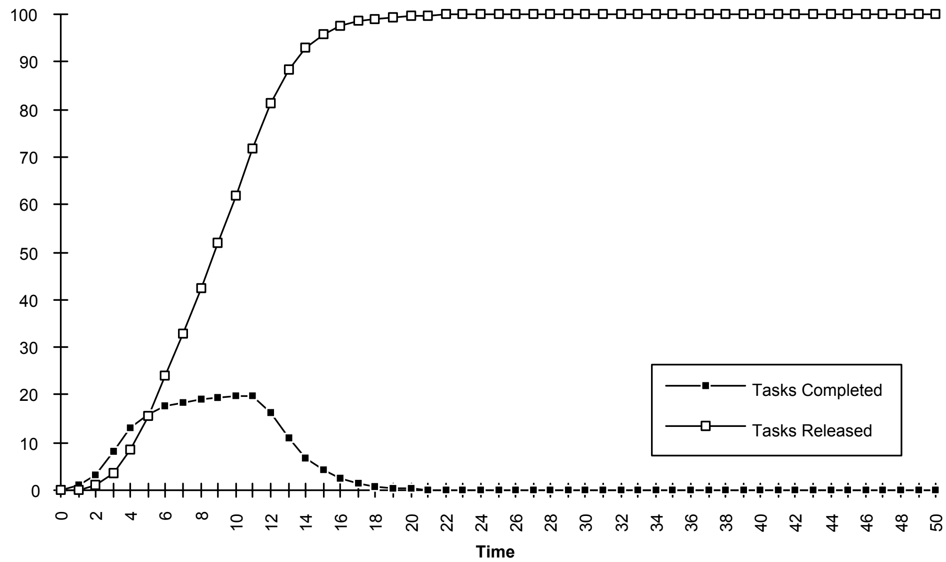

Figure 3-26: Single Phase Simulation with No Errors Development Task Sector Stocks

Some of the impacts of errors can be seen in Figures 3-27 and 3-28 which show the flows and stocks of the development tasks sector with a 50% error generation rate. The flow of tasks due to the discovery of errors are shown in Figure 3-27. This flow responds to the basework rate in the initial peak and stable flows during the middle of the phase duration. The influence of the delays and minimum durations in the system are shown in the variations in the flows at the end of the phase. The stocks which result from the integration of these flows are shown in Figure 3-28. The longer cycle time with errors (46 versus 24) is also evident by comparing Figures 3-25 and 3-26 to Figures 3-27 and 3-28.

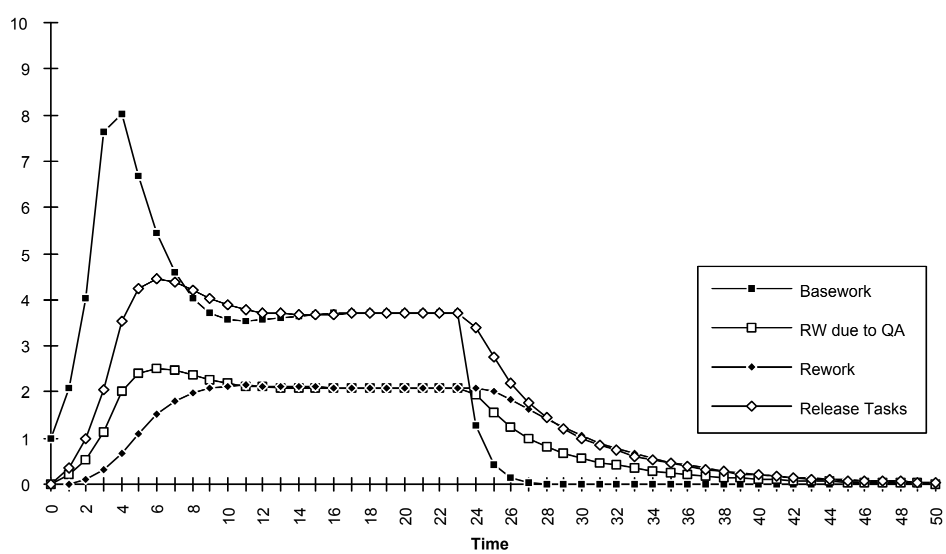

Figure 3-27: Single Phase Simulation with 50% Error Generation
Development Task Sector Flows

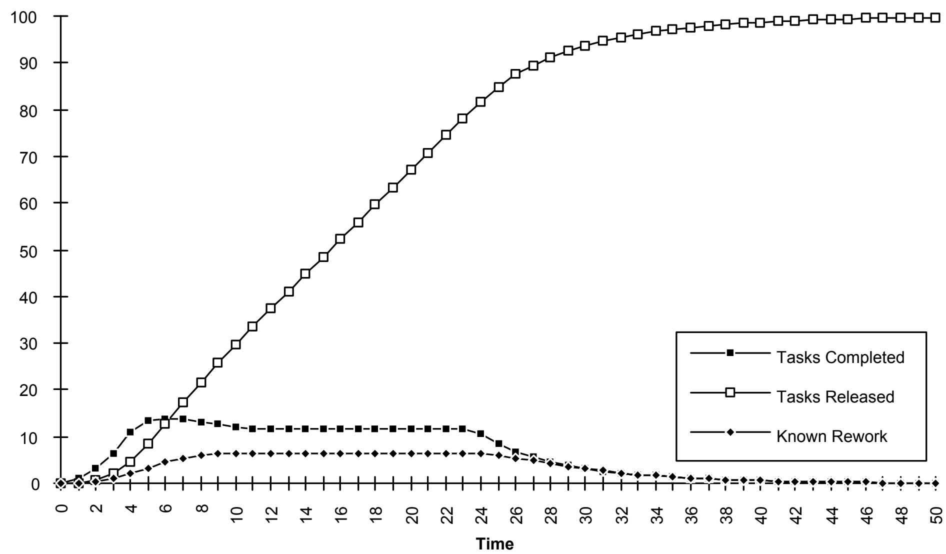

Figure 3-28: Single Phase Simulation with 50% Error Generation
Development Task Sector Stocks

3.4.3 Typical Multiple Phase Model Behavior

A two phase configuration can show some of the important interactions between phases. Figures 3-29 and 3-30 show the flows and stocks for the upstream phase (labeled number 1) of a two phase configuration. Figures 3-31 and 3-32 show the flows and stocks for the downstream phase (labeled number 2) of a two phase configuration. Figure 3-29 shows three of the fundamental flows in the development tasks sector or the upstream phase. The delays seen in the single phase configuration are also evident here. Figure 3-30 also shows that the interaction between the two phases generates a small coordination backlog due to the discovery of errors by the downstream phase which have been released by the upstream phase.

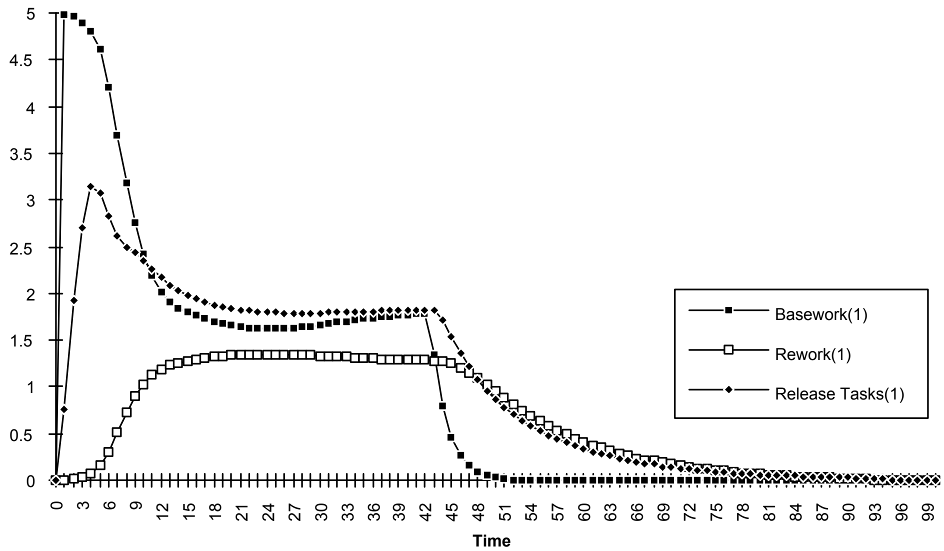

Figure 3-29: Upstream Phase (1) Simulation of Two Phase Model Development Task Sector Flows

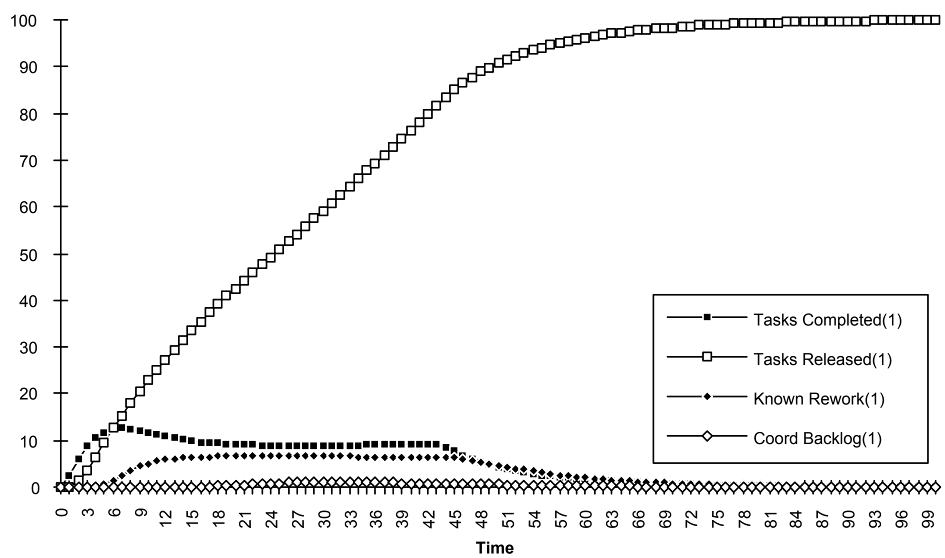

Figure 3-30: Upstream Phase (1) Simulation of Two Phase Model Development Task Sector Stocks

Three of the fundamental development tasks sector flows of the downstream phase are shown in Figure 3-31. The delay caused by the external precedence relationship between the upstream and downstream phases is evident in the start of basework at time 12. The internal characteristics of the two phases are very similar. Therefore the smoothing effect of the upstream phase on the downstream phase can also be seen in Figure 3-31. The resulting lower stock values in the downstream phase are shown in Figure 3-32.

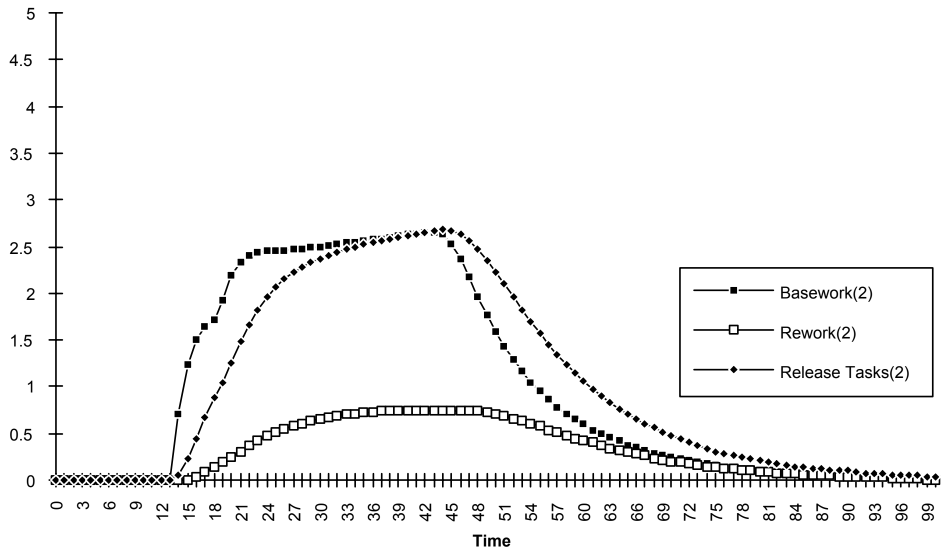

Figure 3-31: Downstream Phase (2) Simulation of Two Phase Model Development Task Sector Flows

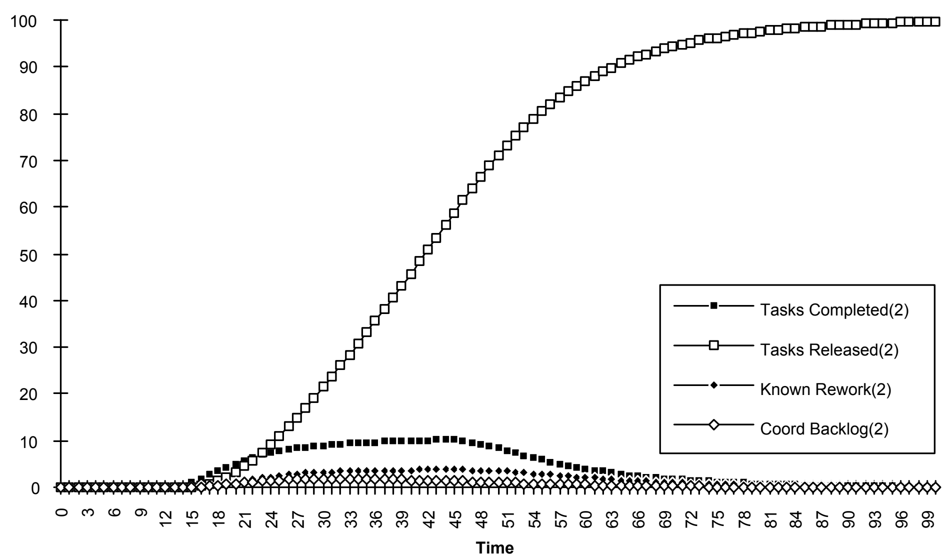

Figure 3-32: Downstream Phase (2) Simulation of Two Phase Model Development Task Sector Stocks

The simulations above show some of the complexity which can be described and simulated with the Product Development Project Model. An understanding of the most influential portions of the model on behavior can be gained from a sensitivity analysis of the model.

3.5 Model Sensitivity to Parameter Values

3.5.1 Introduction

An important part of understanding model behavior is the identification of parameters to which model behavior is sensitive. These parameters can be the focus of parameter estimation work for model calibration and policy and system design. Model sensitivity to parameters is addressed in this section. Model parameter estimation for a specific set of conditions is discussed in Chapter 5: The Python Development Project. The model's sensitivity was tested at the single phase and multiple phase levels.

3.5.2 Parameter Reduction

Sensitivity analysis of the model requires the investigation of variables which are described with a single numerical value at any time (e.g. Minimum Activity Durations) and more complex variables such as Internal Precedence Relationships (Tank-Nielsen, 1980). The number of variables for testing was reduced by eliminating those which do not describe the operation of the real system (i.e. those used for managing the model). Additional reduction was possible by taking advantage of the fact that many parameters are described and used in sets of similar parameters. For example, the time to average actual productivity is used as the basis for a set of four parameters (the four development activities). In these cases the one parameter to which the system is considered most sensitive was kept and the others in the set eliminated for initial sensitivity testing. If initial testing revealed that a parameter set had high leverage the other parameters in the set were tested. The individual members of some sets of parameters was considered too important to apply this reduction (e.g. Minimum Activity Durations for the four development activities). Parameters were also eliminated if another parameter could be used to test the sensitivity of its feedback loop (e.g. delays in reported and expected productivities). Finally, parameters were eliminated if their value was strongly linked to the value of a selected parameter and could be tested through that selected parameter. For example high levels of experience in new team members can be expected to be associated with higher ability to resist schedule pressure. Twenty one parameters remained in a single phase model after these reductions.

3.5.3 Single Phase Model Sensitivity

Sensitivity was tested by observing project performance across a range of parameter values for a hypothetical but internally consistent test development project phase. The test phase is loosely based on the design activity of the project described in Chapter 5. Values for eliminated parameters were established at typical values, defined as a value estimated to be near the mean. Three sets of values were set for the twenty one selected parameters. Each set of values represents a consistent set of conditions. The first set of parameter values represents a pessimistic scenario. A likely scenario estimates the values of a typical project. The third set of values represents an optimistic scenario. The following values were assigned for these parameters for the pessimistic, likely, and optimistic scenarios.

Parameter Name	Pessimistic Scenario Value	Likely Scenario Value	Optimistic Scenario Value
Process Structure and Scope:			
BW_Min_Task_duration	4	2	1
QA_Min_Task_Duration	3	1	0.5
RW_Min_Task_Duration	2	1	0.5
Task_List	1500	1000	500
Internal Precedence Relationship	linear	hyperbolic	open
Basic_prob_flawed_Task	0.9	0.5	0.3
Resources:			
BW_Priority	6	3	1.5
QA_Priority	0.20	1	5
RW_Priority	0.20	1	5
BW_Labor_Delay	6	2	0.5
QA_Labor_Delay	16	8	2
RW_Labor_Delay	4	1	0.25
Max_Headcount	1	3	5
Headcount_Adjustment_Time	16	8	4
Inital_Headcount	0.1	0.5	1.0
Exper_Assim_Time	4	1	0.5
Targets and Performance:			
Initial_Proj_Deadline	25	50	100
Project_Quality_Goal	1.0	0.9	0.7
Project Budget (x 1000)	75	125	250
Quality_Goal_Adjust_Time	48	12	4
Resistance_to_Sched_Slip	0.5	2	5

Table 3-2: Single Phase Sensitivity Test Parameter Values

Sensitivity is measured in the changes in project performance due to changes in parameter values. The three measures of project performance are cycle time, quality and cost. Cycle time is the time required for effectively all tasks (99.99%) to be released. Quality is measured by the number of flawed tasks released. Cost is the total expenses at completion. The test phase performance for the likely scenario is: 55 week cycle time, 363 defects released and $185,000 cost. Model sensitivity is the percent loss or improvement of project performance compared to the performance of the likely scenario due to changing a single parameter's value from the likely scenario value. The raw results of these tests are shown in Table 3-3 below. As an example, when the Quality Assurance Minimum Task Duration parameter is increased from 1 week (likely scenario) to 3 weeks (pessimistic scenario) the cycle time increases from 55 weeks (likely scenario) to 84 weeks (pessimistic scenario). This value is shown in the second column and third row of Table 3-3.

Parameter Name	**Project Cycle Time Performance (weeks)**	**Project Quality Performance (Defects Released)**	**Project Cost Performance ($ x 1000)**
Likely Scenario Performance	55	363	185
Process Structure and Scope:			
BW_Min_Task_duration	67/55	363/398	223/165
QA_Min_Task_Duration	84/47	225/443	238/149
RW_Min_Task_Duration	70/48	364/365	213/168
Task_List	57/49	550/172	194/175
Internal Precedence Relationship	91/58	230/400	233/150
Basic_prob_flawed_Task	98/45	787/215	253/151
Resources:			
BW_Priority	55/55	364/363	185/185
QA_Priority	59/56	376/273	176/192
RW_Priority	55/54	363/387	185/182
BW_Labor_Delay	59/57	303/403	192/166
QA_Labor_Delay	59/49	354/193	223/204
RW_Labor_Delay	55/55	363/363	185/185
Max_Headcount	54/55	367/362	95/243
Headcount_Adjustment_Time	55/55	357/369	231/162
Inital_Headcount	72/54	390/372	129/190
Exper_Assim_Time	55/55	364/364	186/185

Table 3-3: Single Phase Sensitivity Test Results
Project Performance under Pessimistic and Optimistic Scenarios (partial)

Parameter Name	Project Cycle Time Performance (weeks)	Project Quality Performance (Defects Released)	Project Cost Performance ($ x 1000)
Targets and Performance:			
Initial_Proj_Deadline	51/60	339/398	196/161
Project_Quality_Goal	55/55	364/363	185/185
Project Budget (X1000)	55/55	358/363	150/203
Quality_Goal_Adjust_Time	55/55	363/363	185/185
Resistance_to_Sched_Slip	57/60	366/394	181/159

Table 3-3: Single Phase Sensitivity Test Results Project Performance under Pessimistic and Optimistic Scenarios (continued)

The normalized results of these tests are shown in Table 3-4. For example the 29 week loss of schedule performance (84-55=29 weeks) in the previous example represents a 53% increase in cycle time (decrease in performance), as indicated by "-53" in Table 3-3. In a similar manner the "+15" represents an increase in project performance (decrease in cycle time) when the Quality Assurance Minimum Task Duration parameter is decreased from 1 week (likely scenario) to 0.5 week (optimistic scenario).

Parameter Name	Project Cycle Time Performance Change (%)	Project Quality Performance Change (%)	Project Cost Performance Change (%)
Process Structure and Scope:			
BW_Min_Task_duration	-22/0	0/-10	-21/+11
QA_Min_Task_Duration	-53/+15	+38/-22	-29/+19
RW_Min_Task_Duration	-27/+13	0/0	-15/+9
Task_List	-4/+11	-52/+53	-5/+5
Internal Precedence Relationship	-65/+5	+37/-10	-26/+19
Basic_prob_flawed_Task	-78/+18	-117/+41	-37/+18

Table 3-4: Single Phase Sensitivity Test Results Project Performance in Percent Change from Likely Scenario under Pessimistic and Optimistic Scenarios (partial)

Parameter Name	Project Cycle Time Performance Change (%)	Project Quality Performance Change (%)	Project Cost Performance Change (%)
Resources:			
BW_Priority	0/0	0/0	0/0
QA_Priority	-7/-2	-4/+25	+5/-4
RW_Priority	0/+2	0/-7	0/+1
BW_Labor_Delay	-7/-4	+17/-11	-4/+10
QA_Labor_Delay	-7/+11	+2/+47	-21/-10
RW_Labor_Delay	0/0	0/0	0/0
Max_Headcount	+2/0	-1/0	+49/-31
Headcount_Adjustment_Time	0/0	+2/-2	-25/+12
Inital_Headcount	-35/+2	-7/-2	+30/-3
Exper_Assim_Time	0/0	0/0	-1/0
Targets and Performance:			
Initial_Proj_Deadline	+7/-9	+7/-10	-6/+13
Project_Quality_Goal	0/0	0/0	0/0
Project Budget (X1000)	0/0	+2/0	+19/-10
Quality_Goal_Adjust_Time	0/0	0/0	0/0
Resistance_to_Sched_Slip	-4/-9	-2/-9	+3/+14

Table 3-4: Single Phase Sensitivity Test Results Project Performance in Percent Change from Likely Scenario under Pessimistic and Optimistic Scenarios (continued)

The sensitivity of the model behavior is the range of performance change (in percent of likely scenario performance). These results are shown in Table 3-5. For example the 53% decrease in schedule performance and 15% increase in schedule performance for the previously described example produce a 68% total sensitivity of the model's schedule performance to the Quality Assurance Minimum Task Duration parameter. This value is shown in the left data column and second row of Table 3-5.

Parameter Name	Project Cycle Time Performance Range (%)	Project Quality Performance Range (%)	Project Cost Performance Range (%)
Process Structure and Scope:			
BW_Min_Task_duration	22	10	32
QA_Min_Task_Duration	68	50	48
RW_Min_Task_Duration	40	0	24
Task_List	15	105	10
Internal Precedence Relationship	70	47	45
Basic_prob_flawed_Task	96	158	55
Resources:			
BW_Priority	0	0	0
QA_Priority	5	29	9
RW_Priority	2	7	1
BW_Labor_Delay	3	28	14
QA_Labor_Delay	18	49	31
RW_Labor_Delay	0	0	0
Max_Headcount	2	1	80
Headcount_Adjustment_Time	0	4	37
Inital_Headcount	37	5	33
Exper_Assim_Time	0	0	1
Targets and Performance:			
Initial_Proj_Deadline	16	17	19
Project_Quality_Goal	0	0	0
Project Budget (X1000)	0	2	29
Quality_Goal_Adjust_Time	0	0	0
Resistance_to_Sched_Slip	13	7	17

Table 3-5: Single Phase Sensitivity Test Results
Total Sensitivity to Parameter Change

The parameters to which the model behavior is most sensitive depends on the performance measure used. Based on the range of performance changes in Table 3-5, the three performance measures are most sensitive to the following ten parameters:

- **Cycle Time:** Basic probability that a Task is Flawed (96%), Internal Precedence Relationship (70%), Quality Assurance Minimum Task Duration (68%), Rework Minimum Task Duration (38%)

- **Quality:** Basic probability that a Task is Flawed (158%), Task List (105%), Quality Assurance Minimum Task Duration (50%), Quality Assurance Labor Delay (49%), Internal Precedence Relationship (47%)

- **Cost:** Maximum Headcount (80%), Basic probability that a Task is Flawed (55%), Quality Assurance Minimum Task Duration (48%), Internal Precedence Relationship (45%), Headcount Adjustment Time (37%), If the three measures of performance were valued equally (unlikely in practice) the descending order of the four parameters which the model is most sensitive to would be: Basic probability that a Task is Flawed, Quality Assurance Minimum Task Duration, Internal Precedence Relationship, and Maximum Headcount.

3.5.4 Multiple Phase Model Sensitivity

A three-phase model was built to test the sensitivity of the multiple phase model to parameter values. The phase network is shown in Figure 3-33. Six parameters were added to those used to test the sensitivity of the single phase model. Three of those parameters represent the coordination activity, which is not active in a single phase model. These coordination parameters were the Coordination Minimum Task Duration, Coordination Priority and the Coordination Labor Delay. The other three parameters are the three External Precedence Relationships between the three phases in the test model.

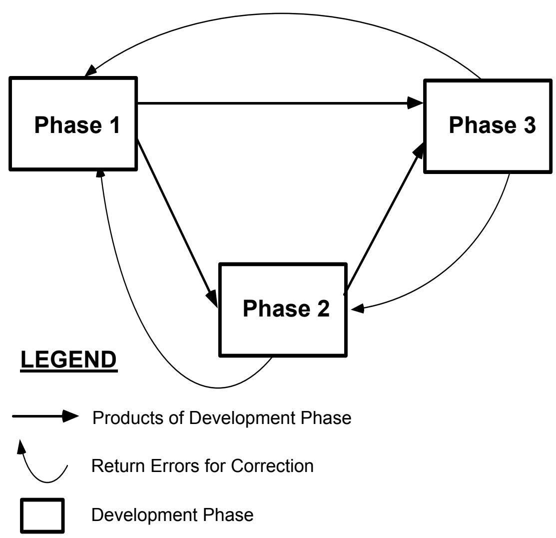

Figure 3-33: Multiple Phase Model for Sensitivity Testing

The sensitivity tests described in the previous section for a single phase test model were repeated for the multiple phase test model using the parameter values shown in Table 3-6. The External Precedence Relationships are shown and described with the description of the Development Tasks sector.

Parameter Name	Pessimistic Scenario	Likely Scenario	Optimistic Scenario

	Value	Value	Value
Process Structure and Scope:			
BW_Min_Task_duration	4	2	1
QA_Min_Task_Duration	3	1	0.5
RW_Min_Task_Duration	2	1	0.5
Coord_ Min_Task_duration	3	1	0.5
Task_List	1500	1000	500
Internal Precedence Relationships	linear	hyperbolic	open
Ext. Precedence Relationship:1-2	sequential	"S"	parallel
Ext. Precedence Relationship:1-3	sequential	lockstep	parallel
Ext. Precedence Relationship:2-3	sequential	"S"	parallel
Basic_prob_flawed_Task	0.9	0.5	0.3
Resources:			
BW_Priority	6	3	1.5
QA_Priority	0.2	1	5
RW_Priority	0.2	1	5
Coord_Priority	0.2	1	5
BW_Labor_Delay	6	2	0.5
QA_Labor_Delay	16	8	2
RW_Labor_Delay	4	1	0.25
Coord_Labor_Delay	8	4	2
Max_Headcount	0.5	1	5
Headcount_Adjustment_Time	16	8	4
Inital_Headcount	0.1	0.5	1.0
Exper_Assim_Time	4	1	0.5
Targets and Performance:			
Initial_Proj_Deadline	75	125	250
Project_Quality_Goal	1.0	0.9	0.7
Project Budget (x 1000)	375	500	1000
Quality_Goal_Adjust_Time	48	12	4
Resistance_to_Sched_Slip	0.5	2	5

Table 3-6: Multiple Phase Sensitivity Test Parameter Values

Three measures of project performance were used in the multiple phase sensitivity tests: total project cycle time, flawed tasks released from the three phases at the completion of the project, and total project cost. The multiple phase sensitivity tests produced the results shown in Tables 3-7, 3-8, and 3-9.

Parameter Name	Project Cycle Time Performance (weeks)	Project Quality Performance (Defects Released)	Project Cost Performance ($ x 1000)
Likely Scenario Performance	140	237	714
Process Structure and Scope:			
BW_Min_Task_duration	154/126	237/239	770/641

QA_Min_Task_Duration	211/125	142/292	882/670
RW_Min_Task_Duration	183/121	225/169	821/660
Coord_ Min_Task_duration	110/168	306/298	591/811
Task_List	141/142	286/168	730/702
Internal Precedence Relationships	177/128	159/289	851/658
Ext. Precedence Relationship:1-2	242/139	178/214	715/689
Ext. Precedence Relationship:1-3	305/131	449/237	894/706
Ext. Precedence Relationship:2-3	320/138	333/202	585/703
Basic_prob_flawed_Task	351/120	434/194	1,011/644
Resources:			
BW_Priority	149/138	253/182	741/704
QA_Priority	142/153	164/284	730/746
RW_Priority	140/136	237/197	714/709
Coord_Priority	140/140	238/239	715/716
BW_Labor_Delay	138/138	303/232	616/714
QA_Labor_Delay	216/79	301/132	985/407
RW_Labor_Delay	141/140	185/242	729/716
Coord_Labor_Delay	163/140	237/237	714/714
Max_Headcount	131/144	122/551	401/1,167
Headcount_Adjustment_Time	139/138	234/221	761/675
Inital_Headcount	141/139	287/229	680/724
Exper_Assim_Time	142/139	243/234	725/709
Targets and Performance:			
Initial_Proj_Deadline	141/142	228/280	725/705
Project_Quality_Goal	140/140	235/237	715/714
Project Budget (X1000)	143/137	273/179	678/740
Quality_Goal_Adjust_Time	140/140	237/237	714/714
Resistance_to_Sched_Slip	141/139	201/311	699/688

Table 3-7: Multiple Phase Sensitivity Test Results
Project Performance under Pessimistic and Optimistic Scenarios

Parameter Name	Project Cycle Time Performance Change (%)	Project Quality Performance Change (%)	Project Cost Performance Change (%)
Process Structure and Scope:			
BW_Min_Task_duration	-10/+10	0/-1	-8/+10
QA_Min_Task_Duration	-51/+11	+40/-23	-25/+6
RW_Min_Task_Duration	-31/+13	+5/+28	-15/+7
Coord_ Min_Task_duration	+21/-20	-29/-26	+17/-13
Task_List	-1/-1	-21/+29	-2/+2
Internal Precedence Relationships	-26/+8	+33/-22	-19/+8
Ext. Precedence Relationship:1-2	-73/+1	+25/+8	0/+3
Ext. Precedence Relationship:1-3	-118/+6	-89/0	-25/+1
Ext. Precedence Relationship:2-3	-128/+1	-40/+15	+18/+1
Basic_prob_flawed_Task	-151/+14	-83/+18	-41/+10
Resources:			
BW_Priority	-6/+1	-7/+23	-4/+1
QA_Priority	-1/-9	+31/-20	-2/-4
RW_Priority	0/+3	0/+17	0/+1
Coord_Priority	0/0	0/-1	0/0
BW_Labor_Delay	1/1	-28/+2	+14/0
QA_Labor_Delay	-54/+43	-27/+44	-38/+43
RW_Labor_Delay	-1/0	+22/-2	-2/0
Coord_Labor_Delay	-16/0	0/0	0/0
Max_Headcount	+6/-3	+48/-132	+44/-63
Headcount_Adjustment_Time	+1/+1	+1/+7	-6/+5
Inital_Headcount	-1/+1	-21/+3	+5/-1
Exper_Assim_Time	-1/+1	-2/+1	-1/+1
Targets and Performance:			
Initial_Proj_Deadline	-1/-1	+4/-18	-1/+1
Project_Quality_Goal	0/0	+1/0	0/0
Project Budget (X1000)	-2/+2	-15/+24	+5/-4
Quality_Goal_Adjust_Time	0/0	0/0	0/0
Resistance_to_Sched_Slip	-1/+1	+15/-31	+2/+4

Table 3-8: Multiple Phase Sensitivity Test Results
Project Performance in Percent Change from Likely Scenario under Pessimistic and Optimistic Scenarios

Parameter Name	Project Cycle Time Performance Range (%)	Project Quality Performance Range (%)	Project Cost Performance Range (%)
Process Structure and Scope:			
BW_Min_Task_duration	20	1	18
QA_Min_Task_Duration	62	63	31
RW_Min_Task_Duration	44	23	22
Coord_ Min_Task_duration	41	3	30
Task_List	0	50	4
Internal Precedence Relationships	34	55	27
Ext. Precedence Relationship:1-2	74	17	3
Ext. Precedence Relationship:1-3	124	89	26
Ext. Precedence Relationship:2-3	129	55	17
Basic_prob_flawed_Task	165	101	51
Resources:			
BW_Priority	7	30	5
QA_Priority	8	51	2
RW_Priority	3	17	1
Coord_Priority	0	1	0
BW_Labor_Delay	0	30	14
QA_Labor_Delay	97	71	81
RW_Labor_Delay	1	24	2
Coord_Labor_Delay	16	0	0
Max_Headcount	9	180	107
Headcount_Adjustment_Time	0	6	11
Inital_Headcount	2	24	6
Exper_Assim_Time	2	3	2
Targets and Performance:			
Initial_Proj_Deadline	0	22	2
Project_Quality_Goal	0	1	0
Project Budget (X1000)	4	39	9
Quality_Goal_Adjust_Time	0	0	0
Resistance_to_Sched_Slip	2	46	6

Table 3-9: Multiple Phase Sensitivity Test Results
Total Sensitivity to Parameter Change

As in the single-phase sensitivity tests, the multiple phase model is most sensitive to different parameters depending on the performance measure used. Based on the range of performance changes in Table 3-, the three performance measures are most sensitive to the following six parameters:

- **Project Cycle Time:** Basic probability that a Task is flawed (165%), External Precedence Relationship: phase 2 to phase 3 (129%), External Precedence Relationship: phase 1 to phase 3 (124%), Quality Assurance Labor Delay (97%), and Quality Assurance Minimum Task Duration (62%).

- **Project Quality:** Maximum Headcount (180%), Basic probability that a Task is flawed (101%), External Precedence Relationship: phase 1 to phase 3 (89%), Quality Assurance Labor Delay (71%), and Quality Assurance Minimum Task Duration (63%).

- **Project Cost:** Maximum Headcount (107%), Quality Assurance Labor Delay (81%), Basic probability that a Task is flawed (51%), and Quality Assurance Minimum Task Duration (31%).

If the three measures of performance were valued equally (unlikely in practice), the descending order of the five parameters which the model is most sensitive to would be: Basic probability that a Task is Flawed, Maximum Headcount, Quality Assurance Labor Delay, two External Precedence Relationships, and Quality Assurance Minimum Task Duration.

3.5.5 Model Sensitivity Test Summary

Sensitivity tests on a single phase test model and a multiple phase model identified parameters which have relatively large influence on model behavior. Model behavior was measured in the three primary dimensions of project performance: cycle time, quality, and cost. All three performance measures are sensitive to the basic rate of error generation, with quality performance being more sensitive than cycle time and cost. This parameter can be seen as a measure of task difficulty or newness to the developers. The high leverage which this parameter has on performance helps explain the challenges of developing new and more complex products.

The parameters to which the model's behavior is most sensitive is relatively consistent between the single phase and multiple phase tests. Approximately half of the high leverage parameters for both the single and multiple phase test models describe the development process. This supports the need for process structure components in dynamic models of projects. All but one of the six performance measures were sensitive to the precedence relationships which describe the levels of process concurrence. This indicates that designing and managing concurrence in development may be a high leverage point for improved performance. Inter-phase relationships overshadowed intra-phase relationships in the multiple phase test. This may indicate that macro-process design and improvement is a higher leverage point than micro-process design and improvement. A second process descriptor, the Quality Assurance Task Duration also influenced five of the six performance measures. This combines with the important role of the Basic probability of a Flawed Task parameter to suggest that quality improvement efforts can be effective at improving development project performance.

In contrast to the process description parameters, none of the high leverage parameters are targets. This implies that setting aggressive project goals is not as effective at improving performance as addressing some other parameters.

Differences in the influence of high leverage parameters on performance measures appear as would be expected. For example Maximum Headcount influenced cost performance more than quality or cycle time. Available work constraints influenced cycle time more often than quality and cost. Quality assurance and rework parameters influenced the quality measure most.

The sensitivity tests identify parameters which deserve particular attention in the estimation of parameter values because of their impacts on simulation results. The results also indicate potentially powerful areas for system design and improvement.

3.6 Model Description Summary

The Product Development Project Model simulates from one to five development phases within a single project. A project phase network links individual development phases through available work constraints and error flows. Targets and performance are measured in three dimensions: cycle time, quality and cost. They are modeled at the project level and also link individual phases. Each development phase models process structure, scope, resources, targets, and performance. The process structure and scope subsystems model development task flows as well as internal, inherited and released errors. Resources subsystem sectors model the quantity, effectiveness and allocation of labor among four development activities. Performance relative to cycle time, quality, and cost targets impact labor allocation, workweek length, and headcount.

Sensitivity tests indicate that model behavior for the three performance measures for single and multiple phase test models is most sensitive to development process parameters, especially task difficulty, precedence relationships and quality assurance parameters. These parameters will be addressed in more detail in Chapter 5: The Python Development Project.