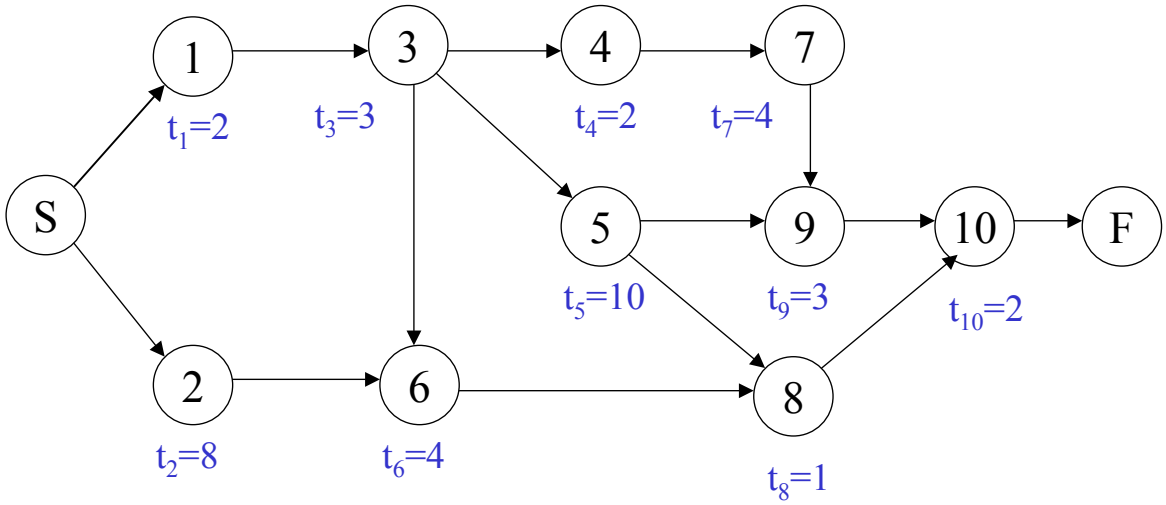
Project Management: Example of Classical Approach

Tennis Tournament Activities (Fitzsimmons, pp 391–392)

Task Description	Code	Immediate Predecessors
Negotiate for location	1	—
Contact seeded players	2	—
Plan promotion	3	1
Locate officials	4	3
Send invitations	5	3
Sign player contracts	6	2,3
Purchase balls and trophies	7	4
Negotiate catering	8	5,6
Prepare location	9	5,7
Tournament	10	8,9

PERT Chart



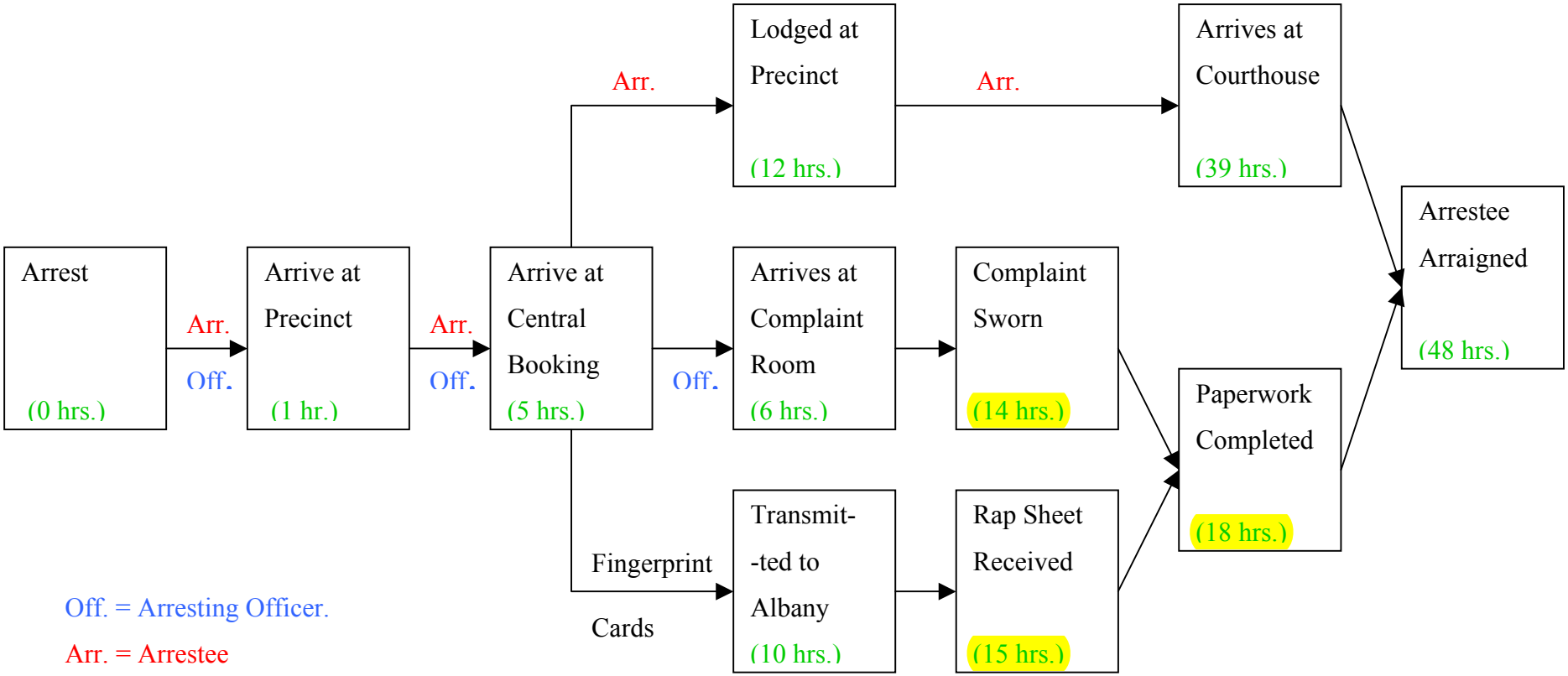
PERT = **P**rogram **E**valuation and **R**eview **T**echnique.

t_i – **completion times** of tasks.

Assume that t_i are **deterministic**.

How to calculate project completion time?

Arrest - to – Arraignment (Larson, ...)



Source:

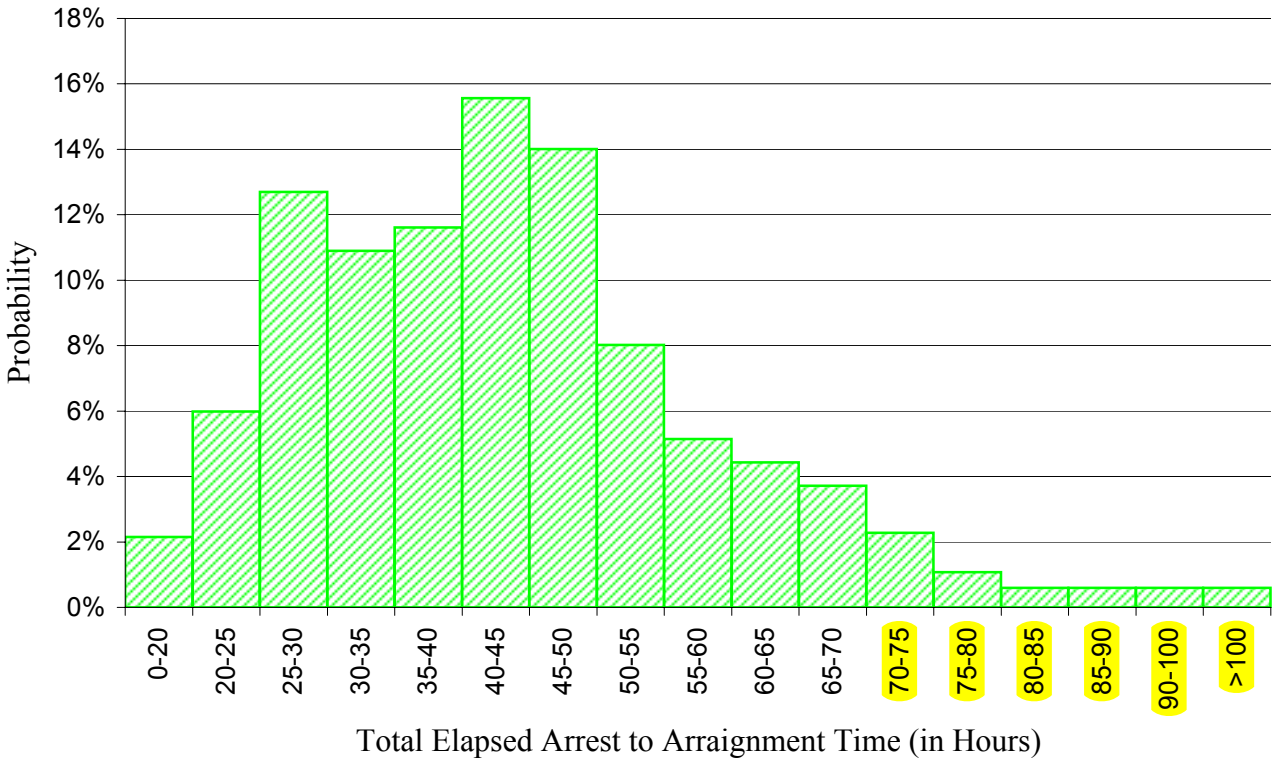
Improving the N.Y.C A-to-A system

Larson, R. Cahn, M. Shell, M.

Interfaces 23, 1993

Arrest to Arraignment Time

Stochastic dynamic model:



Avg. 44.0 hours
Std. 16.2

Should be less than 24 hours.

From Project to Process Management: An Empirically-based Framework for Analyzing Product Development Time

Paul S. Adler[1] • Avi Mandelbaum • Viên Nguyen • Elizabeth Schwerer
Department of Management and Organization, School of Business Administration, University of Southern California, Los Angeles, California 90089-1421
Faculty of Industrial Engineering and Management, Technion, Haifa, Israel 32000
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
Graduate School of Business, Stanford University, Stanford, California 94305-5015

While product development efforts are often viewed as unique configurations of idiosyncratic tasks, in reality different projects within an organization often exhibit substantial similarity in the flow of their constituent activities. Moreover, while most of the planning tools available to managers assume that projects are independent clusters of activities, in reality many organizations must manage concurrent projects that place competing demands on shared human and technical resources. This study develops an empirically-based framework for analyzing development time in such contexts. We model the product development organization as a stochastic processing network in which engineering resources are "workstations" and projects are "jobs" that flow between the workstations. At any given time, a job is either receiving service or queueing for access to a resource. Our model's spreadsheets quantify this division of time, and our simulation experiments investigate the determinants of development cycle time. This class of models provides a useful managerial framework for studying product development because it enables formal performance analysis, and it points to data that should be collected by organizations seeking to improve development cycle times. Such models also provide a conceptual framework for characterizing commonalities and differences between engineering and manufacturing operations.
(*New Product Development; Project Management; Process Management; Development Cycle Time; Processing Network*)

1. Introduction

I'm late, I'm late, for a very important date.
The White Rabbit, *Alice in Wonderland*

This paper presents an empirically-based framework for analyzing product development time in organizations that pursue multiple concurrent nonunique projects using shared resources. Our approach is premised on four hypotheses concerning development activities. First, while product development work is often discussed as if it were composed of idiosyncratic, unprogrammable tasks, in reality many tasks in product development are routine enough to allow their statistical characterization. Second, while projects are often managed as unique configurations of tasks, in reality different projects within a given organization often exhibit substantial similarity in the flow of their constituent activities. Third, while most of the planning tools available to managers assume that projects are independent clusters of activities, in reality many organizations must manage con-

[1] Authors' names are listed in alphabetical order.

0025-1909/95/4102/0458$01.25

Figure 1 **Processing Network Representation**

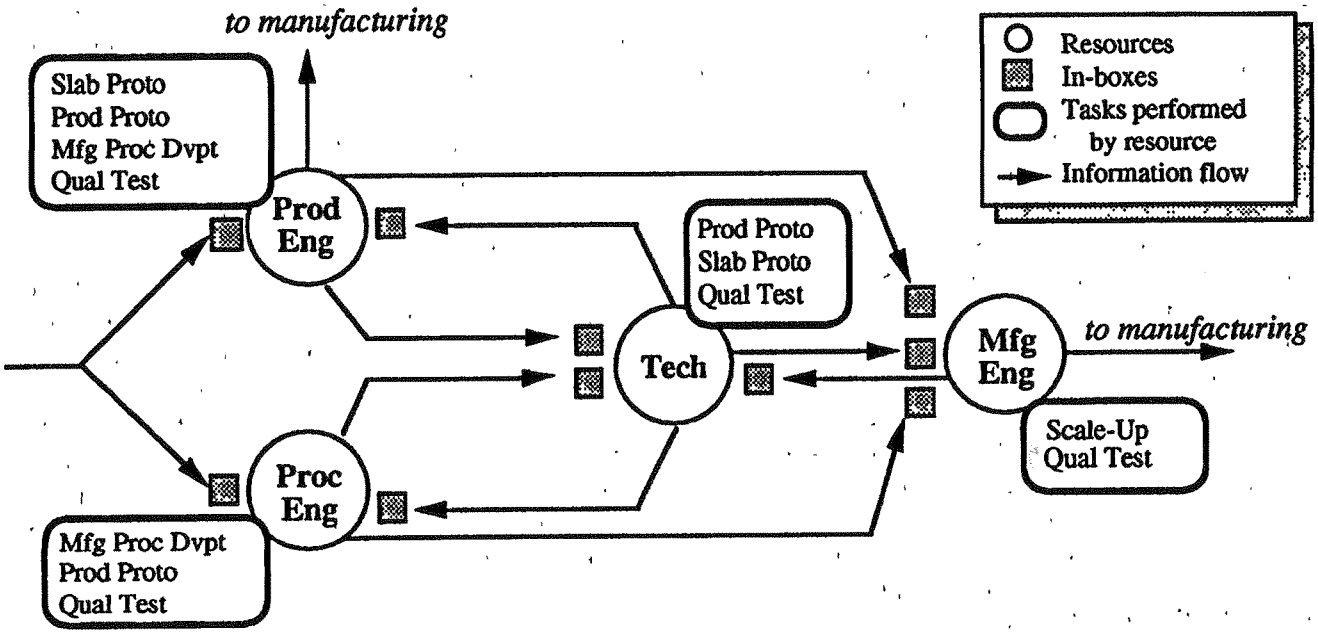
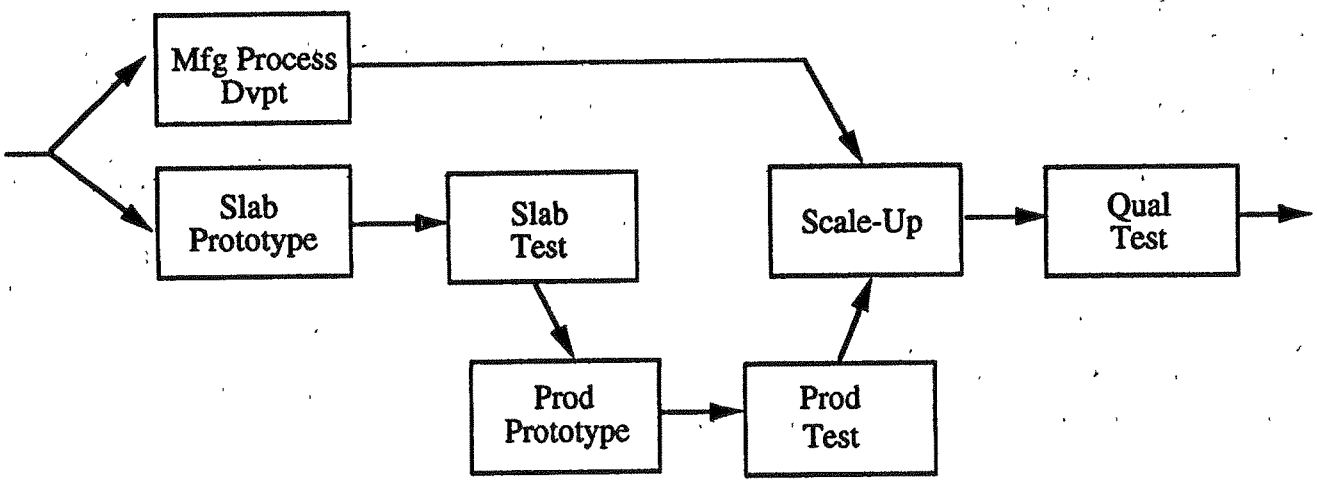


Figure 2 is the PERT diagram associated with the processing network depicted in Figure 1. Each job consists of seven activities. Activities "Manufacturing Process Development" and "Slab Prototype" can be performed in parallel (they represent a fork) and "Scale-Up" begins when activities "Product Testing" and "Manufacturing Process Development" are both completed (a join).

The processing network is stochastic because interarrival times, processing times, and precedence requirements may be subject to statistical variability. Projects are said to be of the same "type" if their individual precedence requirements, processing times, and interarrival times can be characterized by the same set of probability distributions. In the sample organization we

Figure 2 **Traditional PERT Representation**



13

הטכניון – מכון טכנולוגי לישראל
הפקולטה להנדסת תעשייה וניהול
הנדסת מערכות שירות

דוגמא לתרגיל בית:

שרשרת חיול

מגישים :

שרשרת חיול: דוגמא לתרגיל בית

הפרוייקט אותו אנחנו מתארים הינו שרשרת החיול של מתגייס חדש לצה"ל. השרשרת כוללת 13 פעילויות ו תשעה משאבים.
הנחה: התורים במערכת, וזמני השירות הינם סטוכסטיים ואינם קבועים בזמן או בגודל.

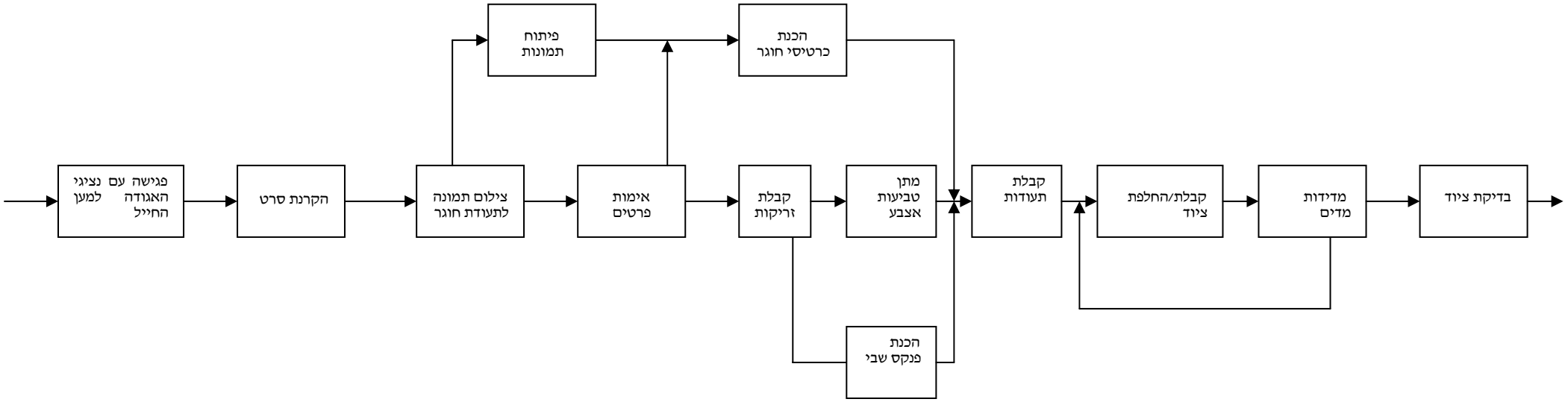
פעילויות במערכת:

- פגישה עם נציגי האגודה למען החייל- בתחנה זו המגייסים החדשים מקבלים שי מטעם האגודה, ומתחילים את שרשרת החיול.
- סרט- המתגייסים צופים בסרט המתאר את שרשרת החיול, ומסביר להם מה צפוי להם בהמשך היום. המתחיילים נכנסים בקבוצה ומקבלים שירות ע"י שרת יחיד.
- צילום תמונה לתעודת החוגר- כל מתגייס בתורו, מצטלם לתמונת הפספורט שתופיע על החוגר. בתחנה זו יש שרת יחיד.
- זריקות- המתגייסים ניגשים לאחות, ומקבלים חיסונים. בתחנה זו נכנסים ביחידות ומשורתים ע"י שני שרתים.
- טביעות אצבעות- המתגייסים מטביעים את טביעת אצבעותיהם על טופס.
- אימות פרטים- המתגייסים נכנסים בתורם לראיון אישי ואימות פרטים אצל מש"קית ת"ש.
- קבלת חוגר- בתחנה זו המתגייס מקבל את תעודת החוגר ואת פנקס השבי שלו. שרת יחיד
- קבלת/החלפת ציוד- קבלת מדים, לבנים וכלי רחצה. בתור זה שני שרתים משרת קבוצה של שלושה אנשים בו זמנית, כל אחד (סה"כ 6 לקוחות בתחנה).
- מדידות- המתגייסים נכנסים בקבוצות של חמישה מתגייסים כל פעם,ומקבלים שירות ע"י חמישה שרתים. (שרת משרת לקוח אחד)
- בדיקת ציוד- בדיקת שלמות הציוד שניתן. שרת יחיד, כ- 20 לקוחות משורתים בו זמנית.
- פיתוח תמונות- תחנה נסתרת למגויס. בתחנה זו מפותחות תמונות הפספורט לחוגר.
- הכנת כרטיס חוגר- תחנה נסתרת למגויס. שני שרתים מכינים את תעודת החוגר, עם התמונה של החייל שצולמה קודם לכן, לאחר אישור לאימות פרטים.
- הכנת פנקס שבי- תחנה נסתרת מהמגויס. שרת יחיד מוציא מהמחשב את פנקס השבי המעודכן לזריקות אותם המגויס ביצע.

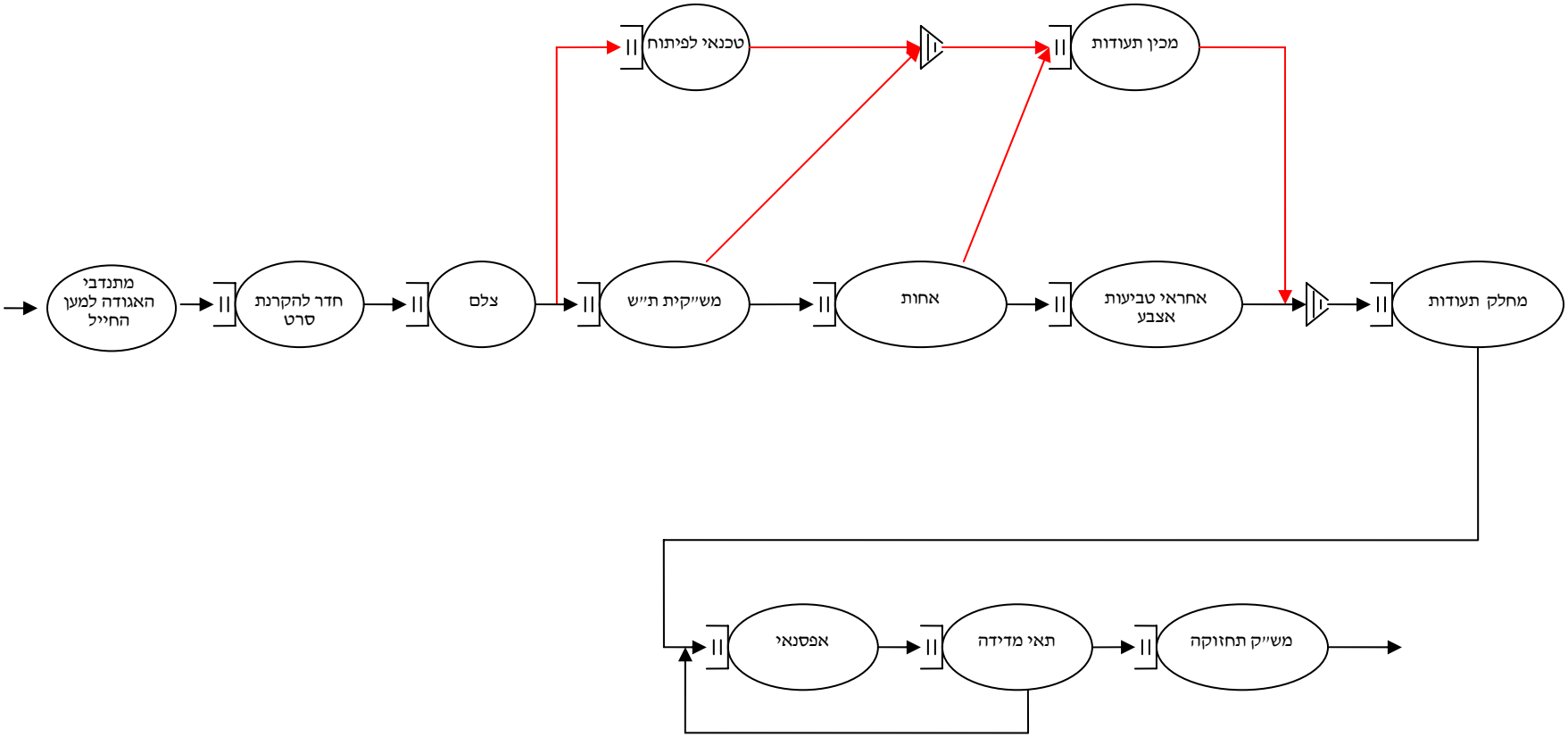
משאבים המערכת:

- מתנדבי האגודה למען החייל- אנשים מחוץ למערכת הצה"לית
- מש"קית ת"ש- אימות הפרטים
- אחיות- למתן הזריקות
- אפסנאי- לתחנת קבלה הציוד
- מש"קי תחזוקה- לתחנת בדיקת הציוד
- צלם- לפספורט
- טכנאי- מפתח את התמונות
- מכין התעודות- הכנת התעודות השונות
- אחראי טביעות אצבע
- מחלק התעודות- מוסר את תעודות החייל למגוייס
- כיתה- להקרנת הסרט
- תאי בדיקת הציוד

שרשרת חיול – דיאגרמת פעולות



שרשרת חיול – דיאגרמת משאבים



מסלול מגויסים — תור סנכרון

מסלול מידע ופריטים נלווים — תור משאבים

סיכום הבעיות במערכת והצעות ייעול

ניתן לראות מהתרשימים כי לפני כל משאב (פרט למתנדבי האגודה למען החייל) קיים תור המתנה. בנוסף, קיימים שני תורי סנכרון במערכת:

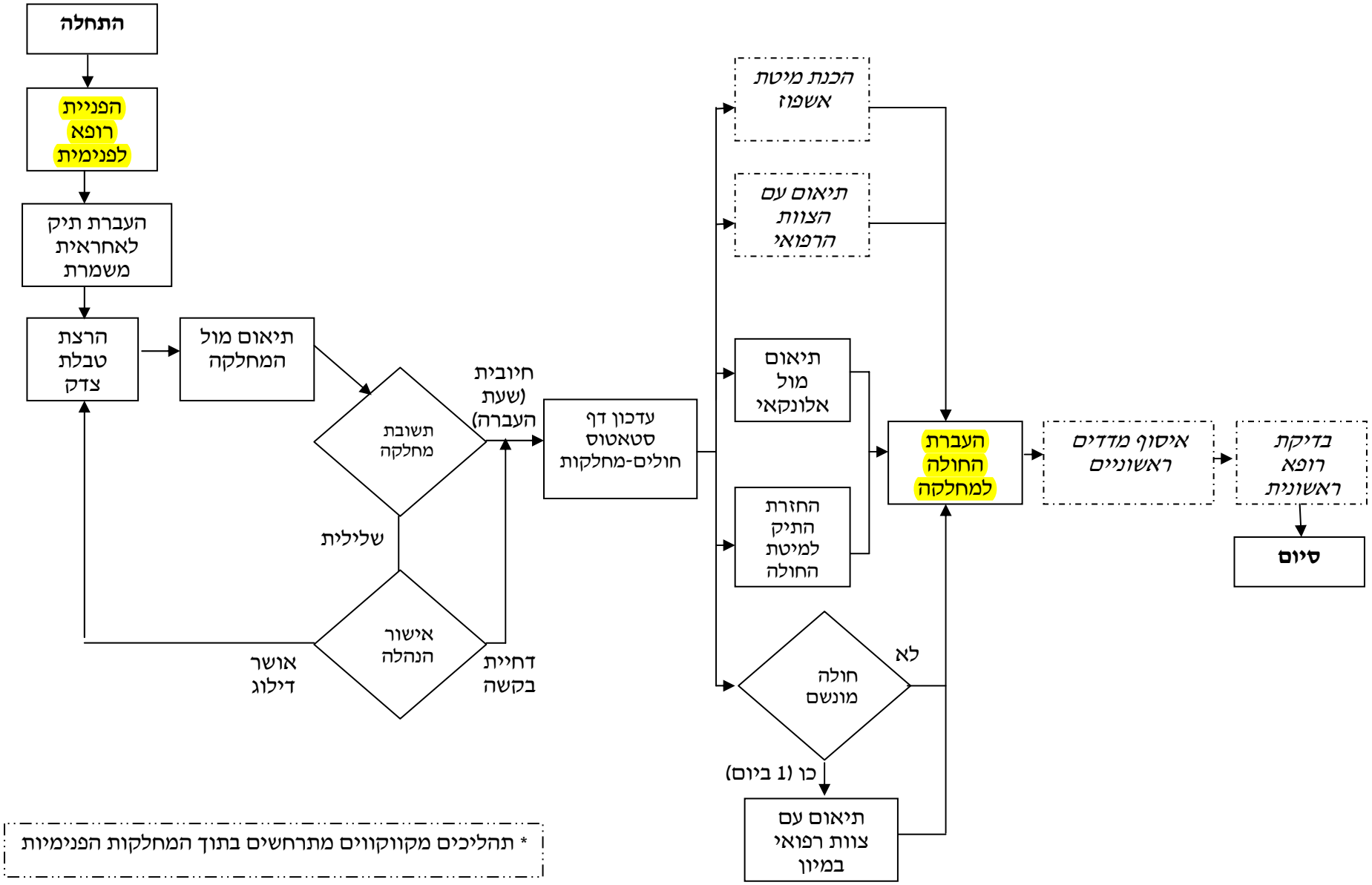
תור סנכרון ראשון לפני הכנת חוגר- אימות פרטים (משיקית תש) ופיתוח תמונות.

תור סנכרון שני לפני מחלק התעודות על מנת לוודא כי שתי תעודות החייל מגיעות לחייל הנכון.

הצעות לייעול המערכת:

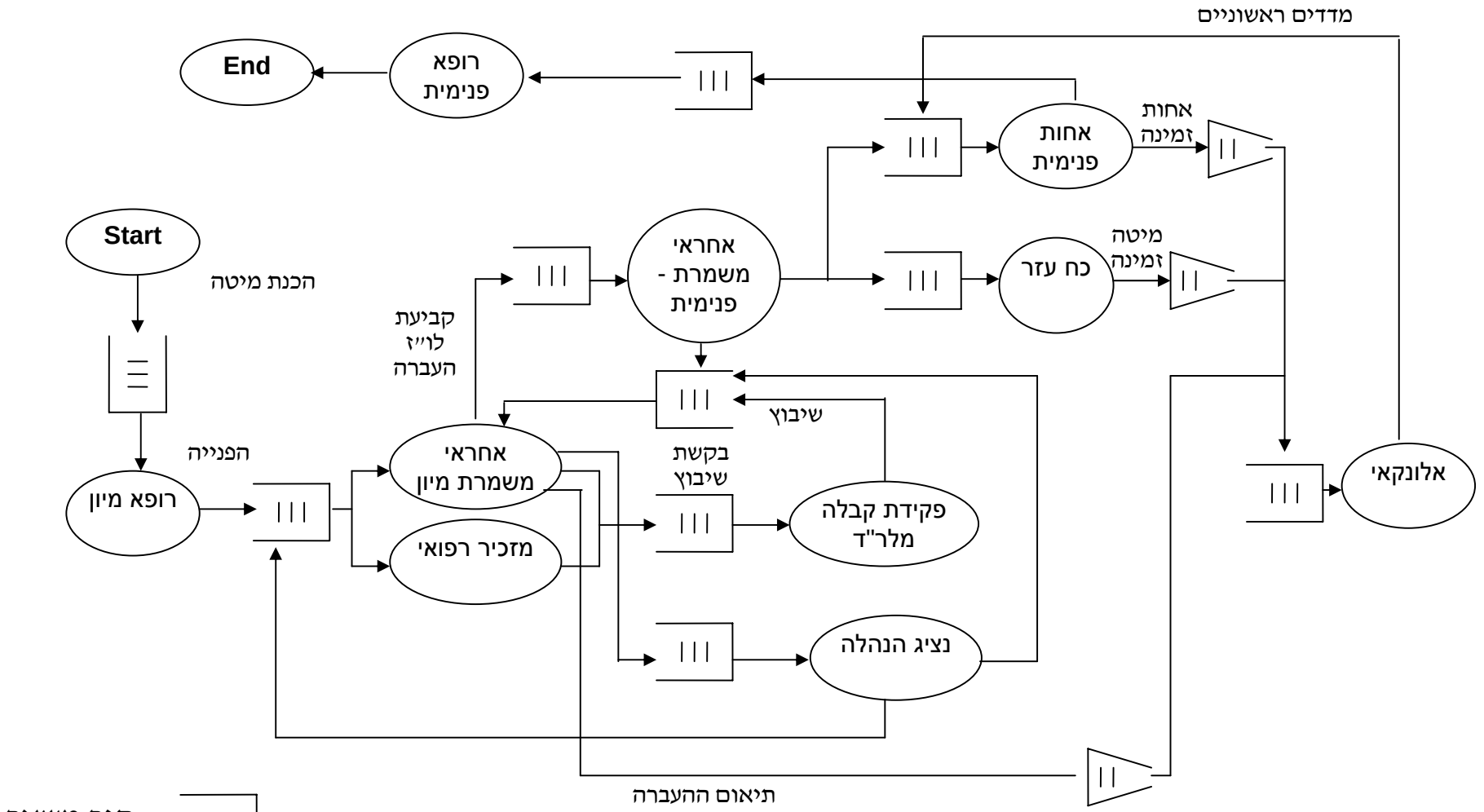
- הוספת שרתים לנקודות המנוקזות לתורי הסנכרון, על מנת ליצור איזון קווים:
  פיתוח התמונות לוקח זמן קצר יותר מהראיון אצל משקית התש. הוספת עמדות תוכל לאזן זאת.
  בתור השני של קבלת התעודות, המסלול של הכנת פנקס השבי מכיל תחנה נוספת בדרך (האחיות), הדבר יוצר עיכוב, ותעודת החוגר "ממתינה" לפנקס השבי. גם כאן הוספת כ"א יכול ליצור איזון קווים טוב יותר, ולגרום לתור הסנכרון להתקצר.
- איזון תורים: השירות בתחנות לא מתבצע באופן זהה. יש תחנות אליהן נכנסת קבוצה ומשורתת בו זמנית (דוגמת תחנות הסרט, מדידת הציוד ובדיקת הציוד), לעומת תחנות בהם החיילים נכנסים ומקבלים שירות אחד אחד. יש לבדוק האם ניתן להפוך את השירות היחידני לשירות קבוצתי (למשל 'שירות עצמי' בתור טביעות האצבעות או קבלת החוגר) על מנת לאפשר שירות גדול יותר של מספר אנשים בעת ובעונה אחת.
- הפיכת תורים סטוכסטיים לדטרמיניסטים:
  תזמון הנכנסים. קביעת זמן חיול לכל נכנס ולא רק תאריך חיול. שליטה על קצב הנכנסים.
- זמני תקן:
  קביעת זמני תקן לפעולות שונות. יצירת בקרה מתמדת ווידוא כי נותני השירות עומדים בזמן זה.
- הפחתת משימות:
  ביצוע חלק מן התהליכים טרם יום החיול (למשל בזמן צו ראשון)- כגון טביעות האצבע, או צילום המתגייס, או הזנת מידותיו לצורך ייעול התהליך בתחנת הקבלה.
- שיפור טכנולוגי:
  העברת הפעולות למחשב- טביעת אצבע ממוחשבת, הקשת פרטים עצמאית (ולא דרך משיקיות התש), בכך יצטמצמו זמני השירות בתחנה.

נספח ח' – תרשים פעילויות וקדימויות



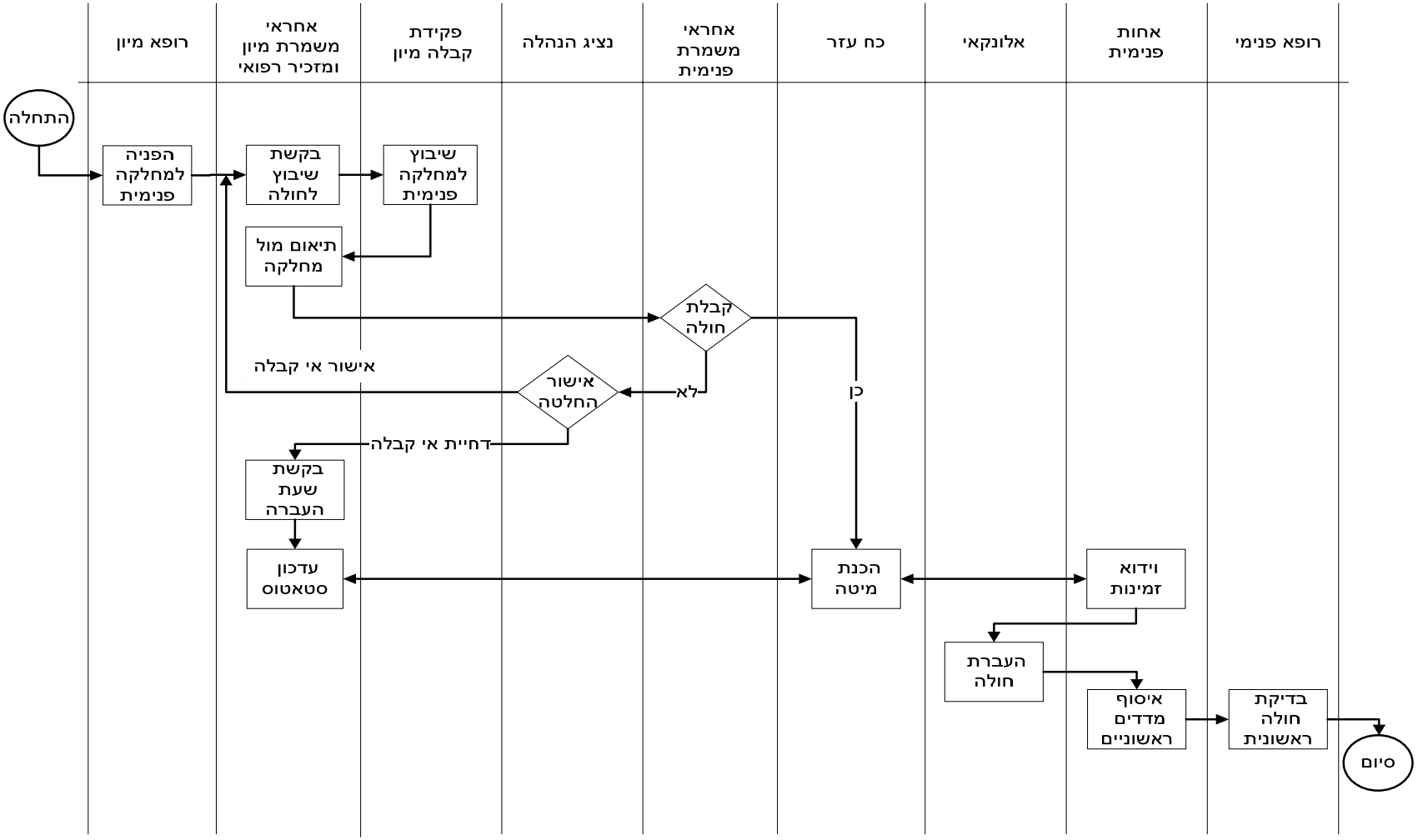
תרשים מספר 6 – תרשים פעילויות וקדימויות

נספח ט' – תרשים רשת משאבים



תרשים מספר 7 – תרשים רשת משאבים

נספח י' – תרשים תהליך משולב



תרשים מספר 8 – תרשים תהליך משולב

Why Queues ?
via Dynamic Stochastic PERT/CPM Networks

- Product/Service development
- Project management

Both "enjoy":

  - Stochastic environment
  - Multi-projects
  - Scarce resources

Service Engineering
Created: July 1996
Last revision: November 2003

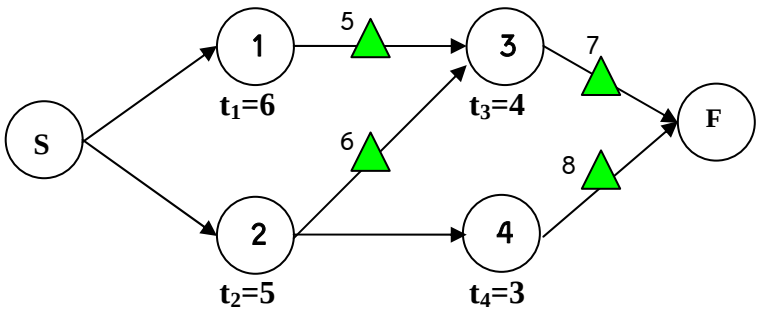
Dynamic Stochastic PERT/CPM Networks

PERT = **P**rogram **E**valuation and **R**eview **T**echnique (**R**esearch **T**ask);
CPM = **C**ritical **P**ath **M**ethod.

Consider a four-task project, whose precedence constraints are expressed by the network diagram below.
The time required for task i is t_i days on average. There are n_i identical "servers" dedicated to task i, and there are many statistically independent replicas of the project to be completed over time.

Model 1: Deterministic PERT/CPM

▲ Synchronization queue



Critical path is S-1-3-F.
Project Completion Time is 10 days.

Model 2: Stochastic PERT/CPM

Warmup model: t_1 = 1 or 11, equally likely, which does not alter given averages.
What is then project duration? How about a 13-days deadline? Critical path?

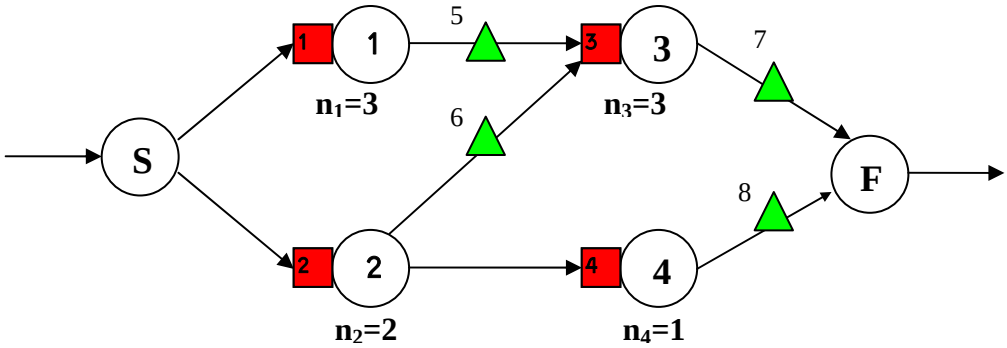
More realistically: Time required for task i is exponentially distributed with mean t_i days and the various task times are independent (random variables). Simulation (spreadsheet) then shows that: Mean completion time = 13.13 days; Standard deviation = 7.36.

Model 3: Dynamic Stochastic Project Networks (PERT/CPM)

New projects are generated according to a Poisson process, the interarrival time being exponential with mean 3.5 days. Each task is processed at a dedicated service station. Tasks associated with successive projects contend for resources on a FIFO basis.
There are two types of queues:
resource-queue: where tasks wait for the resource.
synchronization-queue: where tasks wait until their precedence constraints are fulfilled.

Resource queue (1-4).

Synchronization queue (5-8).



Remark
In general, there could be alternative reasonable definitions of synchronization queues and synchronization times (waiting times in synchronization queues). Such definitions and their interpretations should depend on the particular application in question.
For example, an "Israeli" protocol would specify that if activity 1 is completed before its matching activity 2 then, rather then wait in synchronization queue 5, it immediately joins resource queue 3, waiting there for activity 2 with the hope that it arrives before 1 is admitted to service.

Simulation Description for the Stochastic Models

The behavior of the **static** system was simulated for 50 replications, each with 20,000 projects.
For the static model, project completion time and the distribution of critical path were compiled for each project. (We used 50 replications, each with 20,000 projects, instead of 50*20,000=1,000,000 replications of individual projects, in order to get an approximately normal sample out of the 50 replication means. This is needed to generate confidence intervals, as described further below.)

For the **dynamic** model, the behavior of the system was simulated for 50 replications, 20,000 days each replication.
Data from the first 10,000 days of the simulated operation was discarded, and then summary statistics were compiled for the remaining days of operation.

In both the static and dynamic models, for each project there are three possible critical paths:
s-1-3-f
s-2-3-f
s-2-4-f .
Throughput time, time in queues and critical path were compiled for each project.
Then, for each replication, time statistics were calculated, as well as the distribution of critical paths. These are summarized in subsequent figures, yielding in particular, the mean and std of the throughput time.
At the end of the simulation, standard deviation and confidence intervals were derived, according to the following.

Formulas: ; i- replication index, n= 50.

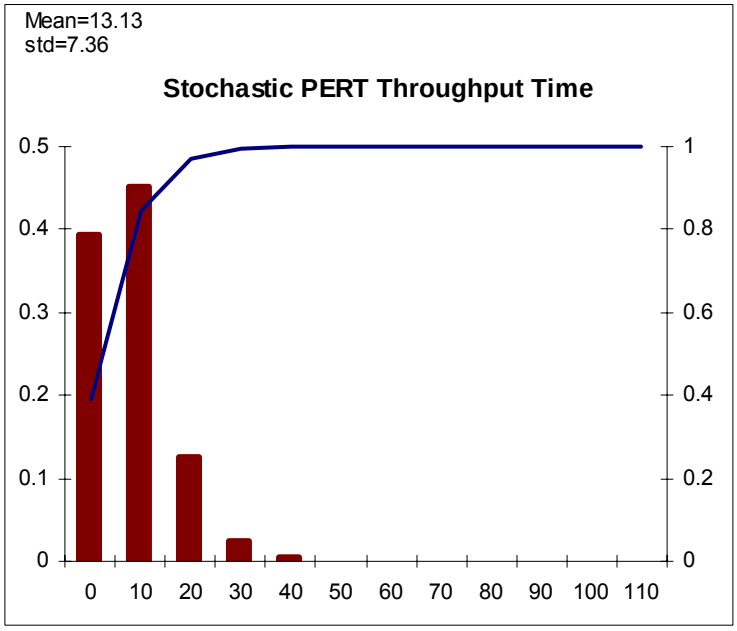
$m = \sum_i x_i / n$; overall mean (x_i - mean of replication i).
$\sigma^2 = \sum_i (x_i - m)^2 / (n-1)$; estimate of variance.
$1-\alpha = 0.95$; confidence level.
$h = t_{n-1,1-\alpha/2} * \sigma / \sqrt{n}$; half-width $1-\alpha$ confidence interval for the mean (based on the normal approximation).

Remark.
The probability that the mean project completion time lies within the interval [m-h, m+h] is equal to $1-\alpha$.

Simulation Results for the Static Stochastic Model

1. Throughput Time

Mean: 13.13 days.
Std: 7.36. Half C.I: 0.095 (0.46% of the mean).



2. Critical Paths

Path	Frequency	half C.I.
s-1-3-f	0.47	0.0074
s-2-3-f	0.26	0.006
s-2-4-f	0.27	0.0058
	1.00	

3. Critical Activities

Criticality index = the probability that a task is on a critical path.

Task	Criticality index
1	0.47
2	0.53
3	0.73
4	0.27

Results for the Dynamic Stochastic Model

1. Capacity Analysis

Question: Can we do it (in steady state) ?
Answer: Calculate servers' utilization ρ, where ρ = λ*E(S) / n.
The answer is NO – We Can't, if some ρ > 1.

λ - rate of new projects. (And also the processing rate at each of the activity nodes!)
E(S) - mean service time of the station.
n - number of (statistically identical) servers at the station.

Servers' utilization (%) = **57,71,38,86**.

2. Response-time Analysis

Question: How long will it take?
Answer: Calculate response/throughput/cycle time.
Present via histograms and Gantt charts.

3. What-if Analysis

Question: Can we do better?
Answer: Sensitivity and parametric analysis.

4. Optimality Analysis

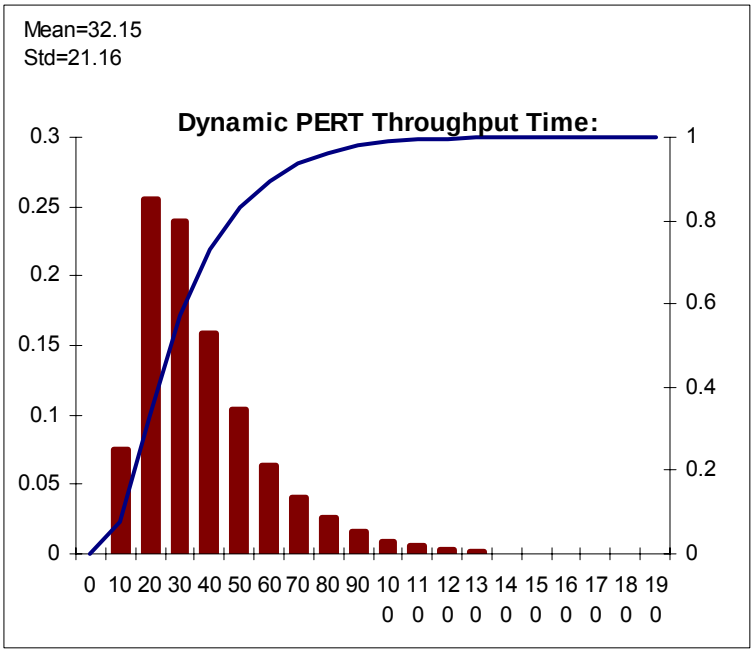
Question: How much better? or: What is the best one could do?
Answer: typically impossible but increasingly possible,
especially in special cases or circumstances.

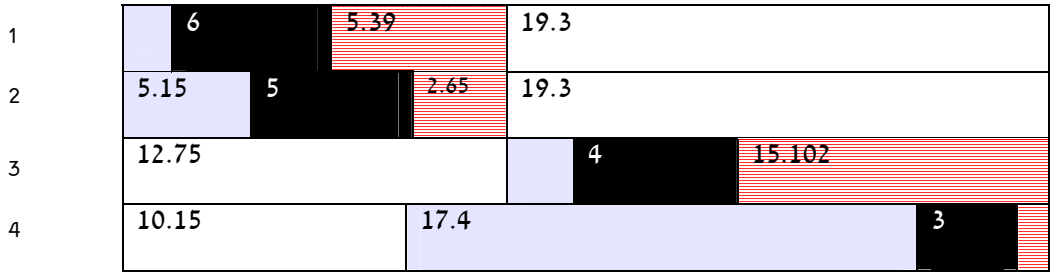
2. Response Time Analysis

How long will it take ? Calculate response/throughput/cycle time.

2.1 Throughput time

Mean=32.15 days.
Std=21.16. Half C.I.=1.5 (4.6% of the mean).
Time Profile: Processing time: 18 days (26.89%).
Waiting time: 24.204 days (36.16%).
Synchronization time: 24.731 days (36.95%).





- waiting time.
- processing.
- synchronization.
- Internal

2.2. Waiting time in queues

Queue	mean	half C.I.	% from mean
1	1.42	0.063	4.43
2	5.15	0.318	6.17
3	0.234	0.009	3.84
4	17.4	1.49	8.5
5	5.39	0.298	5.5
6	2.65	0.076	2.86
7	15.102	1.42	9.5
8	1.589	0.071	4.46

2.3 Critical paths

Path	Frequency	half C.I.
s-1-3-f	0.146	0.0067
s-2-3-f	0.104	0.0052
s-2-4-f	0.750	0.0110
	1.000	

2.4. Critical tasks

Task	Criticality index
1	0.146
2	0.854
3	0.250
4	0.750

Note that task 2 has, by far, the highest criticality index. Yet, task 4 is the clear bottleneck, as far as waiting time is concerned.
The reason for the former is that task 2 participates in “most” paths of the network (2 out of 3).
A reasonable procedure to identify a “critical task” seems to be as follows:
a. Identify the critical path of maximum likelihood (based on 2.3).
b. Identify the task of maximum waiting time (based on 2.2).

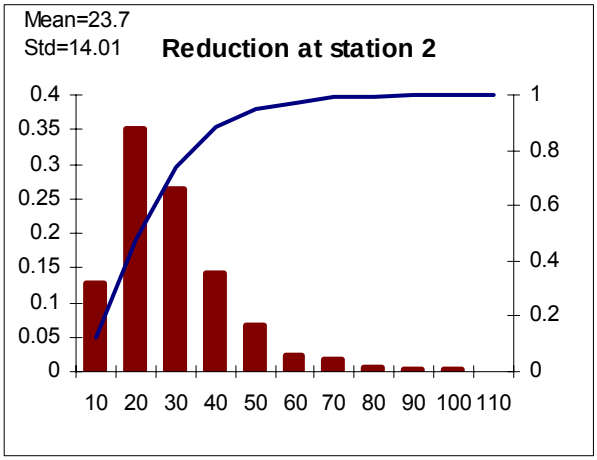
3. What-if Analysis

Question: Can we do better?
Answer: Sensitivity and parametric analysis.

3.1 Reduction at Station 2

Change the mean service time at station 2 to 4 days (instead of 5).
New Mean=23.7 days (improvement of 26.2%).
Std=14.01. Half C.I.=0.368 (1.5% of the mean).
Servers' utilization (%)= 57,57,38,86.

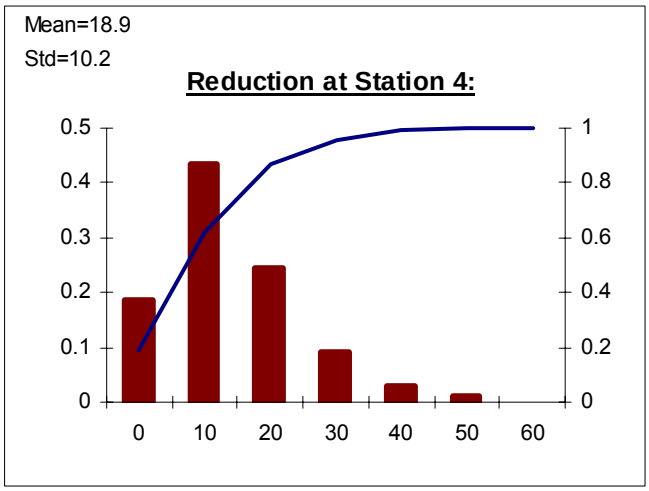
Time Profile: Processing time: 17 days (28.19%).
Waiting time: 21 days (34.82%).
Synchronization time: 22.3 days (36.98%).



3.2 Reduction at Station 4

Change the mean service time at station 4 to 2 days (instead of 3).
New Mean=18.9 days (improvement of 41.2%).
Std=10.2. Half C.I.=0.205 (2% of the mean).
Servers' utilization (%)= 57,71,38,57.

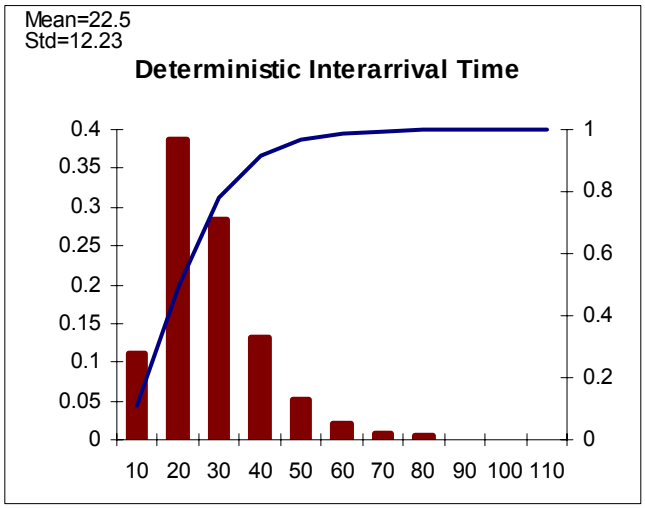
Time Profile: Processing time: 17 days (46%).
Waiting time: 10.6 days (22%).
Synchronization time: 14.5 days (32%).



3.3 Deterministic arrival of projects

Change interarrival time of new projects to exactly 3.5 days (from exponential).
New Mean=22.5 days (improvement of 32.2%).
Std=12.23. Half C.I.=0.63 (2.8% of the mean).
Servers' utilization (%)=57,71,38,86.

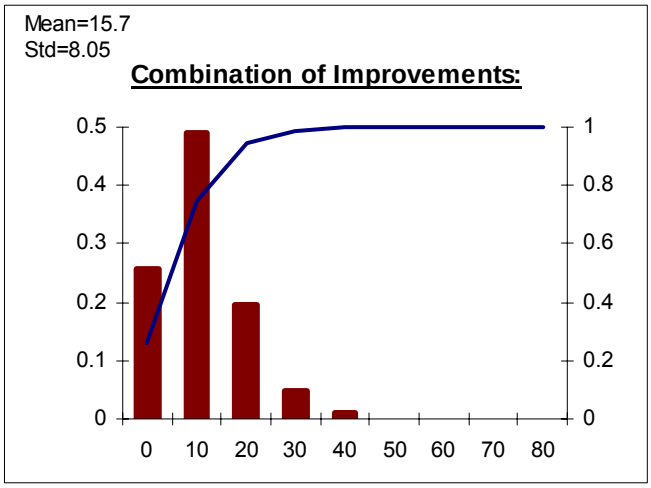
Time Profile: Processing time: 18 days (37.5%).
Waiting time: 12.9 days (26.87%).
Synchronization time: 17.1 days (35.63%).

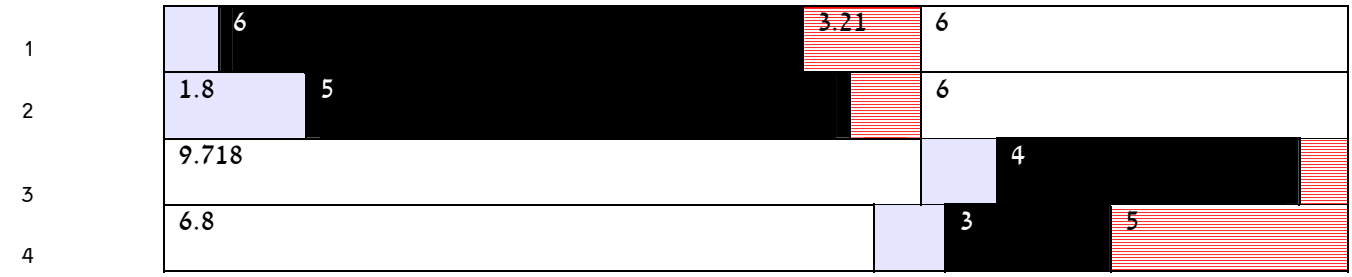


3.4 Combination

Note that a large amount of time is spent at resource queue 4.
Comparing the utilization of station 3 and 4, this suggests a potential process improvement: shift a server from station 3 to 4.
Therefore, the last scenario combines the two improvements: a deterministic interarrival time and shifting one server from station 3 to 4.

New Mean=15.7 days (improvement of 51.16%).
Std=8.05. Half C.I.=0.198 (2% of the mean).
Servers' utilization (%)= 57,71,57,43.
Time Profile: Processing time: 18 days (52.38%).
Waiting time: 3.66 days (10.6%).
Synchronization time: 12.7 days (36.96%).





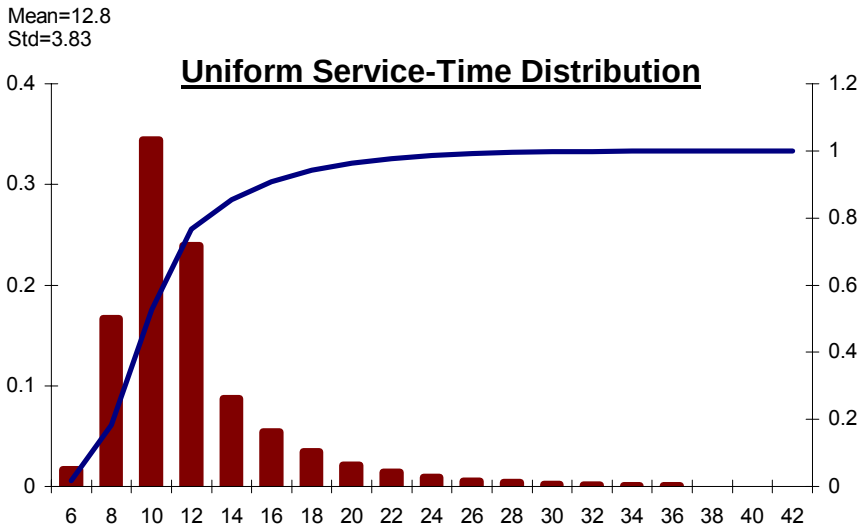
- waiting time.
- processing.
- synchronization.
- internal.

4. Dependence On Distribution

Change, for instance, the distribution of service times from exponential to uniform, but maintain the same mean values as before. Specifically, the time required for task i is uniformly distributed between limits a_i and b_i days, and the interarrival times being uniformly distributed between zero and seven days here.

Task	a_i	b_i
1	3	9
2	3	7
3	3	5
4	2	4

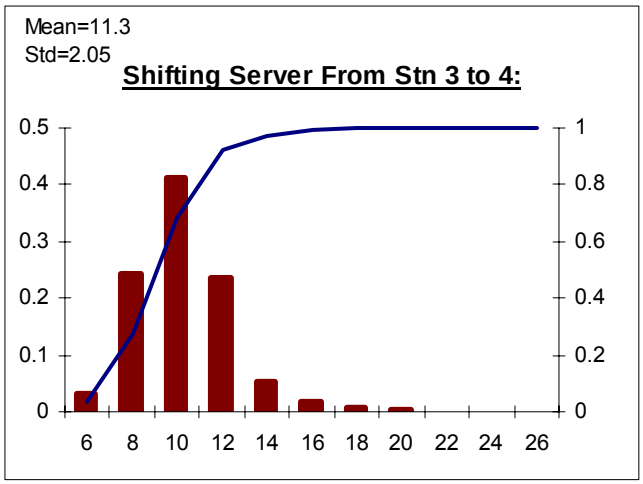
New Mean=12.8 days (Compare with 32.15 days in exponential times).
Std=3.83. Half C.I.=0.034 (0.26% of the mean).
Servers' utilization (%)= 57,71,38,86.



Additional scenarios:

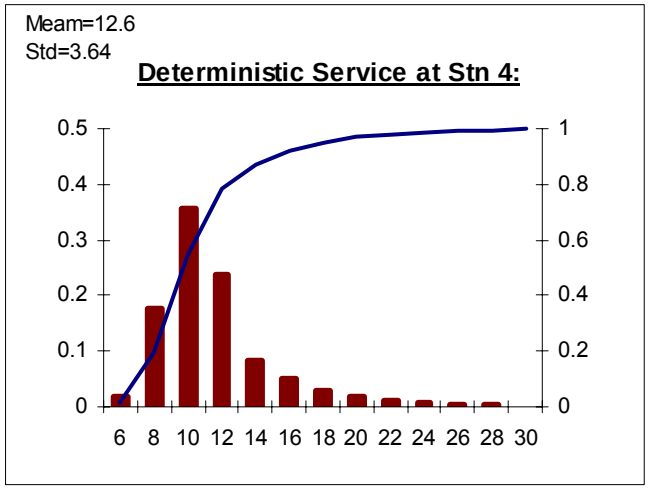
4.1 Allocating Man Power Resources:

Shift a server from station 3 to 4.
New Mean= 11.3 days (improvement of 11.7%).
Std=2.05. Half C.I.=0.017 (0.15% of the mean).
Servers' utilization (%)=57,71,57,43.



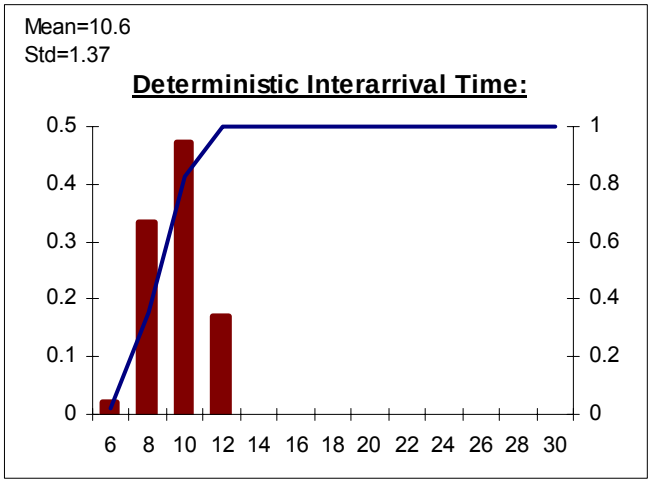
4.2 "TQM" at station 4

Change service at station 4 to deterministic (3).
New Mean=12.6 days (improvement of 1.5%).
Std=3.64. Half C.I.=0.105 (0.83% of the mean).
Servers' utilization (%)=57,71,38,86.



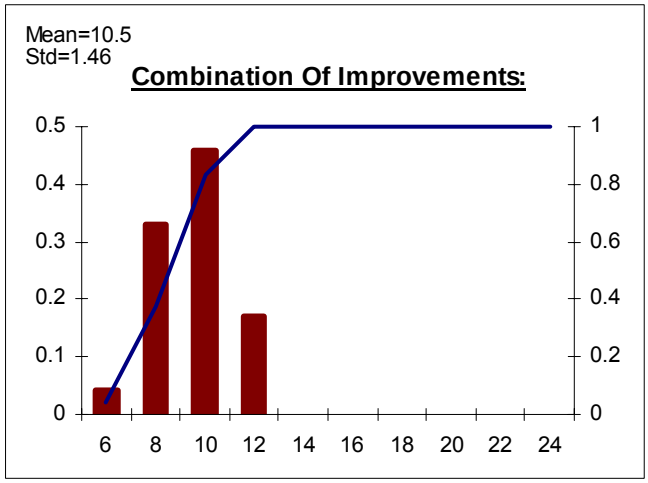
4.3 Deterministic arrival of projects:

Change interarrival time of new projects to exactly 3.5 days.
New Mean=10.6 days (improvment of 17.18%).
Std=1.37. Half C.I.=0.007 (0.066% of the mean).
Servers' utilization (%)=57,71,38,86.



4.4 Combination:

Deterministic interarrival time and shifting one server from station 3 to 4.
New Mean=10.5 days (improvement of 17.96%).
Std=1.46. Half C.I.=0.005 (0.33% of the mean).
Servers' utilization (%)= 57,71,57,43.



5. Dynamic Stochastic Control: Project Management

Consider two types of controls: **open** control, under which all candidate projects are actually initiated, and **closed** where projects must adhere to some predefined criteria in order to be started.
Our open controls are No-Control and MinSLK; the closed control is QSC.
Here are their details:

Open Controls:
1. No Control: A push system with FCFS (First Come First Served) queues. This case was analyzed previously in 2.1.
2. MinSLK: Highest priority in queue to a Minimum Slack activity (MinSLK). Slack-time of an activity is the difference between its Late-Start and Early-Start times. Under MinSLK, as a particular project is delayed then the priorities of its activities increase. Specifically, when an activity of a project is completed, the project's prevalent critical-path is re-evaluated and slack times are updated for the rest of the projects' activities. Then, activities with the least slack time are given the highest priority in resource allocation.

Closed Control:
3. QSC: Queue Size Control (QSC) is based on controlling the resource queue of the bottleneck, the latter being the resource that essentially determines the system's processing capacity. Specifically, one predetermines a maximal number of activities that is allowed, at any given time, within the bottleneck's resource-queue. An arriving project is then allowed into the system to be processed if the length of the bottleneck's resource queue is below this maximal number; otherwise, the arriving project is discarded, never to return (or, alternatively, return late enough so as not to introduce dependencies into the arrival process).

Outline of experiments:

1. **Response time analysis**

2. **The control effect for high throughput rate**

3. **Congestion curves**

5.1. Response time Analysis

How long will it take?
Calculate response/throughput/cycle time.

5.1.1 No Control - see 2.1 on page 6, where we had: Mean = 32.15 days, Std = 21.16.

5.1.2 MinSLK

Mean=21.59 days.
Std=11.57. Half C.I.=0.37 (1.71% of the mean).
Time Profile: Processing time: 18 days (39%).
Waiting time: 12.01 days (26%).
Synchronization time: 16.10 days (35%).

5.1.3 QSC (6)

The maximal number of activities allowed, at any given time, within the resource queue of the bottleneck is 6. (We shall retain this threshold subsequently as well.)

The **bottleneck** resource, namely the resource that determines the system's processing capacity, is taken to be Resource 4. It can be justified by observing that a mere single Resource 4 is dedicated to Task 4; its anticipated utilization level of about 3/3.5 = 86% in steady state, which is by far the highest among all the resources. This choice also finds ample support in our previous analysis (e.g., see 2.2-2.3 on page 7).

λ_{eff} = 0.27 (vs. arrival rate = 0.29 = 1/3.5: 6.9% of the projects are **lost**)

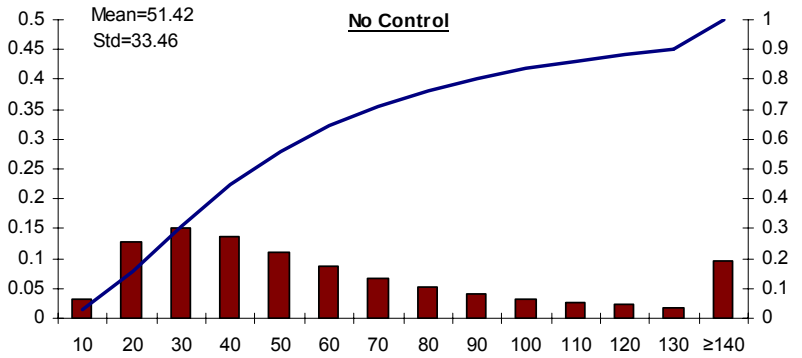
Mean=18.62 days (13.8% lower then MinSLK)
Std=8.80 (23.94% lower then MinSLK). Half C.I.=0.13 (0.70% of the mean).
Time Profile: Processing time: 18 days (54%).
Waiting time: 8.42 days (11%).
Synchronization time: 13.73 days (35%).

5.2. The control effect in heavy traffic

Suppose that the projects arrival rate increases from 1/3.5=0.29 projects/days to an arrival rate of 1/3.25=0.31 projects/days. The load on Resource 4, our bottleneck, increases from 3/3.5=86% to 3/3.25=92%. With this **increased** load, performance is as follows:

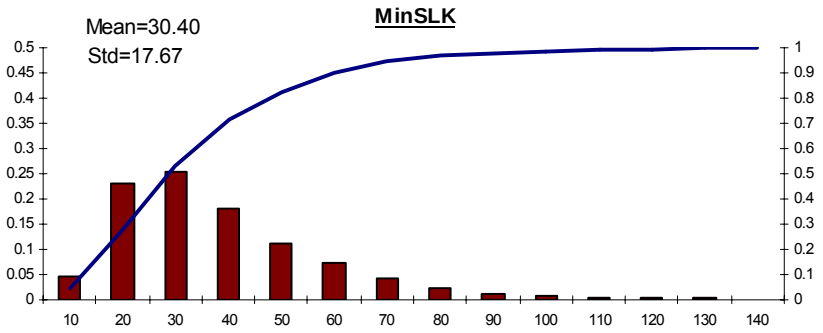
5.2.1 No Control

Mean=51.42 days (vs. 32.15 days in base case).
Std=33.46. Half C.I.=3.86 (7.5% of the mean).
Time Profile: Processing time: 18 days (17%).
Waiting time: 44.83 days (43%).
Synchronization time: 42.42 days (40%).



5.2.2 MinSLK

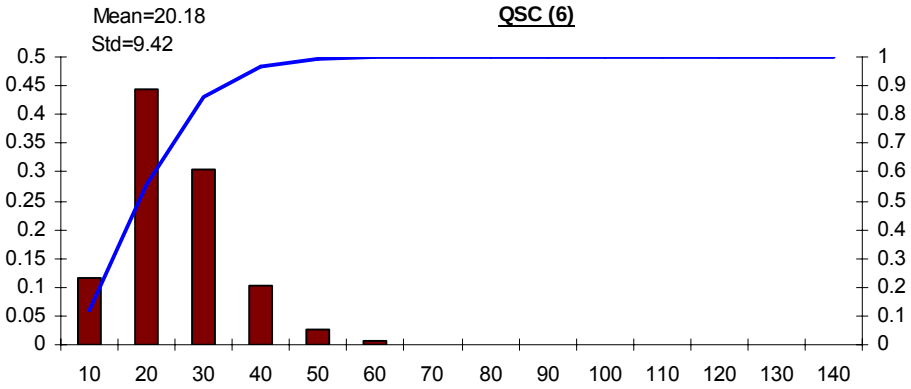
Mean=30.40 days.
Std=17.67. Half C.I.=0.88 (2.9% of the mean).
Time Profile: Processing time: 18 days (27%).
Waiting time: 25.94 days (38%).
Synchronization time: 23.72 days (35%).



5.2.3 QSC (6)

λ_{eff}=0.29 (arrival rate=0.31, 6.4% of the projects are lost)
Mean=20.18 days (33.62% lower then MinSLK; vs. 13.8% under moderate loads.)
Std=9.42 (46.69% lower then MinSLK). Half C.I.=0.19 (0.94% of the mean).

Time Profile: Processing time: 18 days (42%).
Waiting time: 10.47 days (24%).
Synchronization time: 14.67 days (34%).



5.3. Congestion Curves

Now we change the effective throughput rate (x-axis) while recording the mean throughput time (y-axis), for throughput rates between 0.14 to 0.32.
The results, for each of our three controls, are as follows: