Framework for Performance Engineering of Workflows: A Blood Bank Case Study

Gunnar Brataas *, Peter H. Hughes † and Arne Sølvberg ‡
Department of Computer and Information Science
Norwegian University of Science and Technology (NTNU)
7034 Trondheim, NORWAY

Abstract

An integrated approach to performance analysis in terms of throughput and response time to satisfy customer requirements is particularly important for workflow systems where human and computerised processes are intertwined. A basic framework for performance engineering is proposed, which bridges the gap between the design of computerised information systems and of organisations in which they are embedded. The framework is based on the Structure and Performance specification method (SP), and is illustrated through the presentation of a blood bank information system.

1 Introduction

Organisations increasingly depend on computerised workflow systems which assist case workers in delivering services to customers. An integrated approach to performance analysis of workflows in terms of throughput and response time to satisfy customer requirements is needed [17]. The total information system, consisting of the workflow system and the case workers, must have acceptable *performance* in terms of response time, throughput, and utilisation of the available resources. *Performance engineering* of information systems aims at developing systems with acceptable performance [16]. Performance engineering has until now almost exclusively dealt with the performance of computerised information systems. This paper addresses performance engineering of the total information system consisting of both the computerised workflow system and the case workers who use it. For the organisation, this paper relies on a mechanistic view [12]. A comprehensive theory of organisational performance must also take other views into account, e.g. the social view.

This paper builds on a thesis where the performance engineering method outlined below was explored via two case studies [1].

I. Specify system requirements:

- **IA.** Determine objectives of performance model.
- **IB.** Specify system boundaries.
- **IC.** Estimate workload.
- **ID.** Determine performance requirements.

II. Create performance model:

- **IIA.** Establish static model.
- **IIB.** If necessary, establish dynamic model.

III. Guide system development.

IV. Verify and refine overall performance model.

One of the two case studies, the *The Gas Sales Telex Administration Case Study* has been presented in [1, 3], and the other case study, namely the *The Blood Bank Case Study* (BLOODCASE) is described in this paper.

The blood bank system and the performance problems encountered, provided a motivation for seeking a single framework where both organisations and computer based aspects of system performance could be combined. This was described at a conceptual level in [2]. In this paper we revisit BLOODCASE, focusing on step IIA of the method, static modelling, and particularly on the interaction between case workers and the computer system.

The rationale for static modelling is described in Section 2. BLOODCASE is introduced in Section 3. The original transfusion process of BLOODCASE is described in Section 4, and the computerised transfusion process in Section 5. Performance problems in the manual subsystem is investigated in Section 6.

*Present affiliation: Telenor R & D, Kongens gate 8, 7005 Trondheim, Norway. Email: Gunnar.Brataas@fou.telenor.no
†Also at: Modicum Ltd, Congleton, Cheshire CW12 3HZ, United Kingdom. Email: phh@modicum.demon.co.uk
‡Email: Arne.Solvberg@idi.ntnu.no

A conceptual framework which governs the approach, the *basic framework* is introduced in Section 7. The underlying modelling paradigm *Structure and Performance* is described in Section 8 and illustrated with respect to BLOODCASE. Section 9 discusses similarities between human and computerised resources, while differences are discussed in Section 10. Section 11 offers some conclusions.

2 Rationale for Static Modelling

Models of performance usually focus on the dynamic behaviour of systems, using techniques such as queuing network analysis [9] or discrete-event simulation (e.g. [5]) to examine the effect on system response-time and capacity of different design alternatives and workloads. This dynamic modelling depends upon the availability of quantitative estimates of the demand placed on each resource of the system by each task. The resource demands are used as input parameters to the dynamic model, which then computes the effects of contention for resources among competing elements of the workload.

In this paper, the focus is on the step which precedes dynamic modelling, namely the calculation of resource demand. This is often known as static modelling. The reason for this focus is three-fold: Firstly, resource demands set the upper bounds on the performance a given system can deliver. Secondly, as system development becomes more compartmentalised, with for example the performance of an application depending upon several layers of intervening software components, appropriate representation of system structure for static modelling becomes technically demanding and requires methodological support. This is especially true where mixed human and computer information systems are involved. Thirdly, for the type of problem considered, the techniques of dynamic modelling are well understood, and may be applied in a routine way. However the accuracy of the dynamic model depends critically upon the accuracy of the resource demands which are its input parameters.

SP is a static modelling paradigm which enables us to relate overall resource demand to the properties of the system and its components in a very general way [8]. It therefore assists the calculation of the global effects of local changes. Because of its emphasis on system structure, SP can be readily combined with other formal methods of system description, used in the design process [13]. In BLOODCASE the PrM method was used, as described in Section 3.

3 The Blood Bank Case Study (BLOODCASE)

BLOODCASE comprised performance engineering of a transaction-oriented information system which was developed for a major hospital in Norway. Process models were specified in the experimental CASE tool PPP [7] and were annotated with parameters describing estimated use of software resources, showing the practical feasibility of a method for using performance engineering tools integrated with common CASE tools [1]. While there were no performance problems in the computerised parts of the blood bank system, severe problems were discovered in the interaction between humans and computers.

Figure 1 depicts a simplified process modelling (PrM) view of the blood bank. PrM which is part of the PPP experimental case tool, and is an extension of data flow diagrams (DFDs) [7], which specifies processes and their interaction in a formal way. Triggering and termination of processes are specified using ports, making the semantics formal. The interaction between processes is specified using flows, which is extended to denote control flow and material flow as well as dataflow. No distinction is made between flows carrying data and flows carrying material.

The basic human roles in the blood bank are donors, patients and laboratory engineers. The blood bank consists of the four main processes, donor administration, blood stock administration, transfusion administration and laboratory administration:

Administrate Donors Assisted by laboratory engineers, donors give raw blood which are stored in the blood bank. The blood bank is the name of the cabinet where all the different blood products are stored.

Administrate Blood Stock Makes sure the stock of blood will meet future demands for blood. Blood is also purchased from other blood banks.

Transfusion After testing, blood products are transfused to patients.

Laboratory Performs several tests on blood. Raw blood consists of several blood products which are grouped according to the AB0 and the Rh test system. These test systems reduce the risk of giving patients blood products which do not match the patients blood type, which would usually have fatal consequences.

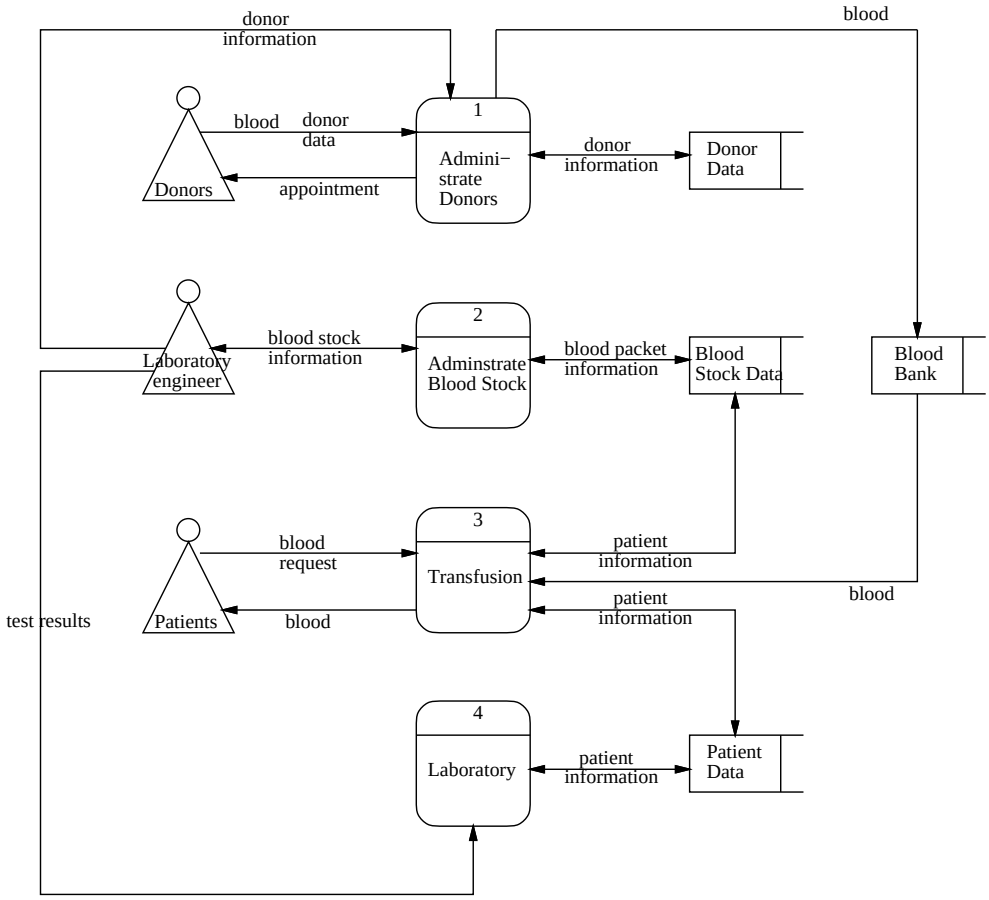

Figure 1: Top level blood bank model. These four information subsystems can be decomposed into 34 processes [6].

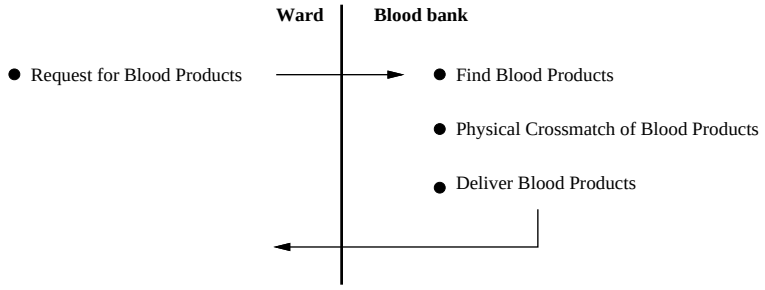

Figure 2: The manual transfusion process in the blood bank organisation.

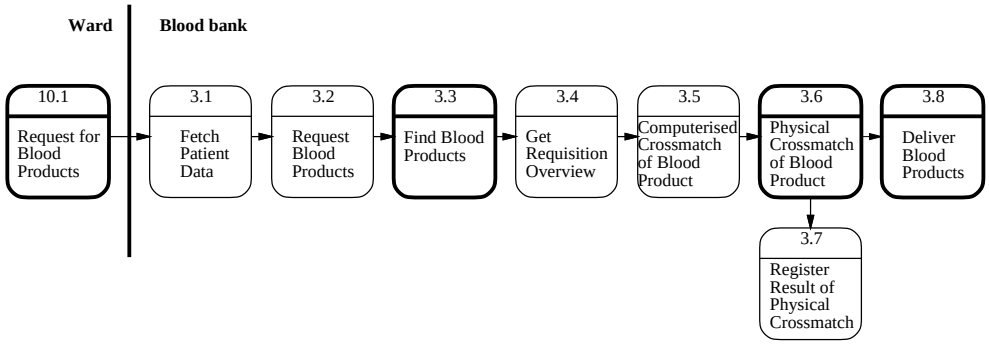

Figure 3: The transfusion process in the blood bank as it was implemented. Manual processes are indicated by **boldface** process symbols.

4 The Original Manual Transfusion Process

In the rest of the presentation we focus on the transfusion process where performance problems in the interaction between humans and the computer were discovered. The transfusion process takes care of the transfusion of blood from the blood bank to patients in need for blood. Figure 2 shows informally the main (sub)processes of the transfusion process, and is a decomposition of process 3 in Figure 1. The processes of Figure 2 are carried out by humans, and the relevant processes are:

Request for Blood Products The ward of the patient requests one or more blood products for the patient from the blood bank. This request is brought to the blood bank organisation on a sheet of paper.

Find Blood Product The blood bank organisation finds suitable blood products in the physical blood bank.

Physical Crossmatch of Blood Products Laboratory engineers mix a sample of donor blood with patient blood, looking for indications of mismatch.

Deliver Blood Products Blood products are delivered to the ward and the transfusion process is terminated as seen from the blood bank organisation's point of view (given that no complications arise).

5 The New Computerised Transfusion Process

To improve the efficiency of the blood bank, and to ease the interaction between the patient ward and the blood bank, it was decided to develop a blood bank information system. To the transfusion process five computerised (sub)processes were added, which also brought about changes to the human (sub)processes. The computerised transfusion process is depicted in Figure 3. Process **Find Blood Products (3.3)**, **Physical Crossmatch of Blood Products (3.6)**, and **Deliver blood products (3.8)** are done by humans without computer support. The other process components are:

Fetch Patient Data (3.1) Patient data is fetched from the blood bank computerised information system.

Request Blood Products (3.2) Blood products are requested based on information from the ward. The information on the paper with the request from the ward must here be typed into the computerised information system.

Get Requisition Overview (3.4) A laboratory engineer investigates the requisitions made for a patient

Computerised Crossmatch of Blood Product (3.5) A laboratory engineer performs a computerised crossmatch between the donor blood and patient blood.

Register Result of Physical Crossmatch (3.7) Invoked for each blood packet requested for the patient.

6 Performance Problems in the Manual Subsystem

BLOODCASE was done to investigate the modelling of computer performance problems. The study showed that there were no problems with the computer capacity, but after the system was put in operation, a severe performance problem was discovered in the interaction between the computer and the staff of the blood bank [2]. The introduction of the computerised information system actually slowed down the performance of the organisation. This experience changed the direction of the research into developing an approach for performance modelling that takes human aspects and computer aspects into consideration within the same modelling framework. To understand the problems with the transfusion process, this section first introduces some background material about both the manual and the computerised version of the transfusion process. A solution to the problems is then presented in Section 8 which is based on the performance engineering method of Brataas' thesis [1]. The problem processes are:

Get Requisition Overview (3.4)
The computerised process took approximately 1 minute to execute, and was therefore never used. Part of the reason for this long time was that the patient's name had to be reentered, even if it was already present in the system. The system developers had not properly understood the workflow, and they did therefore not discover that process 3.4 would succeed process 3.3, which makes reentering of data unpractical.

Computerised Crossmatch of Blood Product (3.5)
This process was not performed as prescribed, because it was much more convenient to look directly into the physical blood bank than to type in the necessary data and wait for the answer from the computerised information system. Instead, requests for blood packets were handwritten in a book, and fed into the computerised information system after the whole process had been terminated. Whenever the laboratory engineers wanted to look something up, this book was more convenient to use than the computerised information system.

Register Result of Physical Crossmatch (3.7)
This process also created problems, because the process had to be performed once for every packet of blood which was requested for a patient. When up to 10 blood packets for one patient were needed in a hurry, this was not very practical, and created another argument for the informal book-based approach.

The basic problem in these three cases was the slow interaction between the manual and the computerised information system. If too much data had to be entered or if too many screen images had to be invoked, this would sometimes slow down the workflow and hence the performance of the organisation.

Accuracy of the information in an information system is important. Therefore, it is more critical if *update* operations are neglected, compared to *retrieve* operations. Thus, there are no serious problems if computerised retrieve operations like process **Get Requisition Overview (3.4)** and process **Computerised Crossmatch of Blood Product (3.5)** are not used, but replaced by some manual processes. However, this could have been discovered by including manual work for entering data in a performance model used during development of the computerised information system. But the process **Register Result of Physical Crossmatch (3.7)**, which *updates* information is more important. This process will therefore be considered more closely in Section 8.

The information system processes (i.e. the composite manual/computerised processes) were not explained in the information system documentation, and this made it very hard to get an overview of the relations between the different screen images which were used to enter and display information. It required careful study of the original documentation to find implicit information which could give some clues to understand the total information system processes. In addition, interviews had to be performed. If the information system processes had been recorded in the first place, understanding the transfusion process would have been easier.

Also training of personnel contributed to the decrease in efficiency of the computerised transfusion process. The transfusion process may be improved by entering patient data directly from the ward. Actually, this was also implemented with the computerised information system, but sufficient training of the ward personnel had not been done [15]. An information system is an interaction between humans and the computerised information system, and is not stronger than the weakest link in the chain, in this case training of personnel.

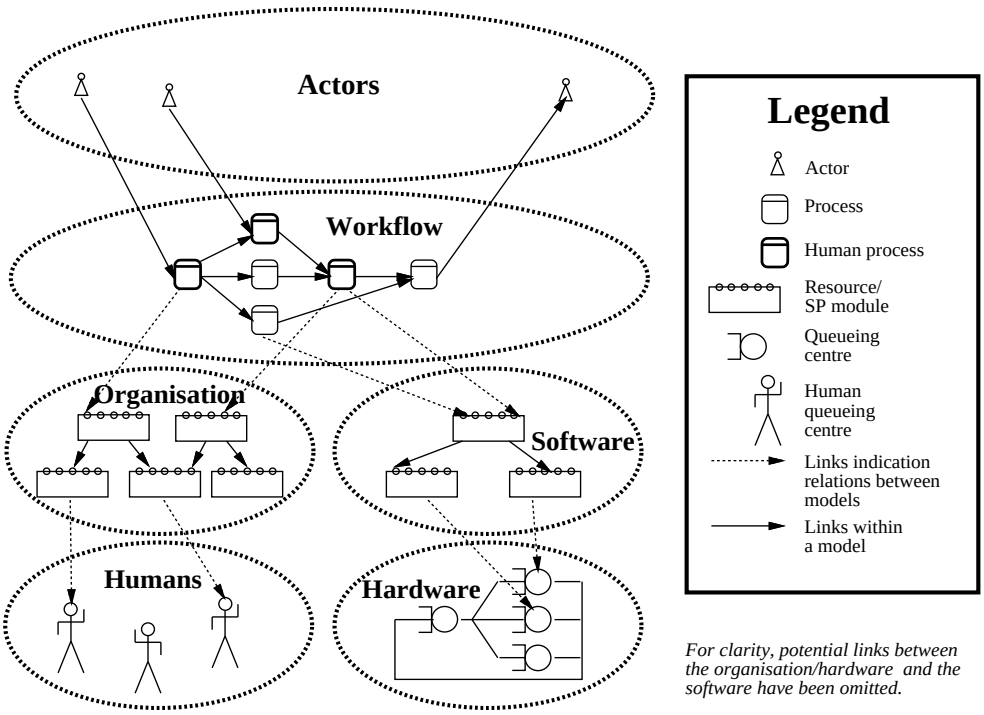

Figure 4: Basic framework for performance engineering in workflow organisations.

7 Basic Framework

A workflow system handles the accesses to workflows, and typically contains the four basic elements [11]: (1) *Workflows* to reach business goals. (2) *Human resources* performing workflows based on business rules. By definition, workflows involve more than one person. Roles which are independent of specific people are important for workflow, by increasing the flexibility. (3) *Computerised resources* invoked by the human resources for executing workflows. (4) *Data* accessed by the tools. The same data may be used by several computerised resources. The same computerised resources may also be used by several human resources, and the same human resource may be used for several workflows.

Workflows is often centred around (electronic) documents which flow between the environment and the organisation and within the organisation. Probably the most basic difference between workflows is the degree of predictability: (1) *routine workflows* have deterministic resource demands, (2) *template workflows* have statistical resource demands, e.g. some subprocesses may be optional, and (3) *ad-hoc workflows* have unpredictable resource demands, often because the next subprocess is not determined before the current subprocess is finished.

As illustrated in the basic framework in Figure 4, *workflow processes* use *organisational resources* like departments and sectors which again use *humans*. Workflow processes also use *software resources* like databases or file systems which finally use *hardware resources* like disks, networks and CPUs. Note that the term resource is used more narrowly than the ordinary use of this term. In this paper, a resource is reactive. A resource carries out work which is initiated by an actor. *Actors* start workflows and use the results of workflow. Thus humans have a dual role in this framework. A human will often by and actor. On the other hand, humans may also act as part of organisational resources which may perform work. A clear distinction cannot be made between actors and resources, because a person or a computer hardware subsystem offering operations to other actors, and therefore acting as a resource, may also become an actor when demanding work from other resources. An actor may also execute part of his own workflow, and will then become a resource. Viewing a person or hardware subsystem as an *actor* or a *resource* depends on the abstraction level.

In the figure, the solid lines inside of each ellipse describe relationships between the objects in each ellipse, whereas the dotted lines describe relationships between objects in different ellipses. The six ellipses in Figure 4 together constitute a *workflow system*.

Each resource offers *operations* to resources at higher levels of abstraction and uses operations from resources at lower levels of abstraction. Resources can either be *existing* or *projected*. In a performance model, the properties of a resource are described via parameters. For existing resources, measurements may be used to provide these parameters. For projected resources, estimation is necessary.

8 Structure and Performance (SP)

SP is a method for describing the hierarchical and modular nature of a system and for analysing the resource requirements of its component parts [8, 18]. In SP a resource which provides a related set of services or *operations* is known as a *component*. More formally, a component implements a data type, i.e. a data structure defined by the operations which can be performed upon it. Until now, SP has almost exclusively been used to model *computerised* information systems. To illustrate SP, we introduce a model of a very simple *manual* information system, the blood bank application in Figure 5.

As described in Section 6, the process **Register Result of Physical Crossmatch (3.7)** was critical. Bad performance in this process is particularly critical when several blood packets are needed in a hurry. The need for blood packets may be described by the operation `get_blood_product` in the blood bank application in Figure 5. This operation is indicated with a circle in the `Blood bank application` resource. For this `get_blood_product` operation, resource consumption for both the `Blood bank computer system` and the manual information system with the `laboratory engineer` must be specified. We will focus on the manual resource `laboratory engineer` in this paper, because it was the human interaction (with the computer system) which was problematic, and not the computer system. The performance for the blood bank computer has been estimated in [1].

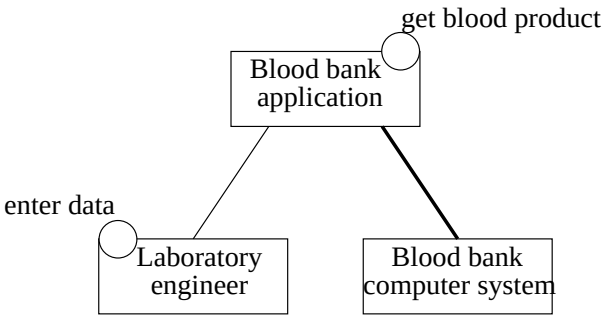

Figure 5: A sketch of the resources needed for earlier prediction of potential problems. The legend is in Figure 4.

Operations in SP are typed. The basic types of operations in SP are processing operations, memory operations and communication operations. Typing of operations in SP is indicated by the thickness of the links in SP models as shown in Figure 5. Processing operations have solid lines, memory operations have bold lines and communication operations have dotted lines.

8.1 Complexity Specifications

Each link between two components is represented by a *work complexity specification matrix*. An example is shown below:

$$C^{Blood\_bank\_application}_{Laboratory\_engineer} = \begin{matrix} \\ get\_blood\_product \end{matrix} \begin{matrix} enter\_data \\ \left[\begin{array}{c} 10 \end{array}\right] \end{matrix}$$

In this example, the component `blood_bank_application` with the operation `get_blood_product` devolves work on the component `laboratory_engineer` with the operation `enter_data`. This complexity specification corresponds to the link between these two components in Figure 5. The number 10 in this simple matrix means that *on the average*, there has to be 10 `enter_data` operations for each `get_blood_product` operation.

The work devolved by `blood_bank_application` to a component at a lower level, e.g. `laboratory_engineer` is calculated by adding the work for all possible paths between the two components. The work along each path is calculated by multiplying the complexity matrices for each link along the path.

In this case there is only one link. More complex examples are shown in [1, 3], and as a representative example we show the complexity specification between the `server` and the `diskh` (disk handler) for the blood bank computer system [1]:

$$C^{server}_{diskh} = \begin{matrix} \\ COBOL \\ select \\ select\_list \\ update \end{matrix} \begin{matrix} \begin{array}{cccc} COBOL & sql\_msg & read\_block & write\_block \end{array} \\ \left[\begin{array}{cccc} 1 & & & \\ & 1 & 1 & \\ & 6 & 44 & \\ 1 & & 1 & 1 \end{array}\right] \end{matrix}$$

The `server` component offers the three SQL database operations (singleton) select, select list and update, and in addition offers COBOL statements for processing. These four `server` operations devolve work on the four disk handler operations COBOL (statement), SQL message, read block and write block. In this complexity specification it is for example shown that the operation `select_list` on the average uses 6 `sql_msg` operations and 44 `read_block` operations.

8.2 Resource Demands

The mapping between the devolved work on a component and the time required to carry out the operations is specified by a set of resource demand vectors. As an example of resource demands, the resource `laboratory engineer` may offer the operation `enter_data`, which may take 20 seconds, or 0.33 minutes:

$$D_{Laboratory\_engineer} = enter\_data \left[\begin{array}{c} 0.33 \end{array} \right]$$

When a complexity specification is combined with a resource demand vector, the total resource demands are calculated:

$$D^{\text{Blood\_bank\_application}}_{\text{Laboratory\_engineer}} = C^{\text{Blood\_bank\_application}}_{\text{Laboratory\_engineer}} \cdot D_{\text{Laboratory\_Engineer}}$$

Using the earlier expressions for $D^{\text{Blood\_bank\_application}}_{\text{Laboratory\_engineer}}$ $C^{\text{Blood\_bank\_application}}_{\text{Laboratory\_engineer}}$, we get:

$$D^{Blood\_bank\_application}_{Laboratory\_engineer} = get\_blood\_product \left[\begin{array}{c} 3.3 \end{array} \right]$$

Thus, entering data 10 times, one time for each of the 10 packets of blood, will result in a response time of about 3.3 minutes, when contention is not considered. A contention model could easily have been introduced if necessary.

The distinction between work and load is important in SP. *Work* is defined as a vector $\vec{w}$ of operations applied on a given component or resource in SP. For work, there is no information about time: how long time an operation takes or the sequence of operations. *Load* may be modelled as a transaction rate λ, for an open system, or as a concurrency N, for a closed system, where N represents the number of continually active processes, e.g. user terminals. Taken together, work and load define the workload for some system. If for example five patients need blood packets, this workload may be expressed as:

$$W^{Blood\_bank\_application}_{Today,Normal} = \begin{array}{c} get\_blood\_products \\ \left[\begin{array}{c} 5 \end{array} \right] \end{array}$$

When a load model and a work model are combined with a contention model, we have a performance model. In this case, the interaction only with the computer system will take around $5 \cdot 3.3 = 17$ minutes. With such response times, we obviously have performance problems.

9 Similarities between Human and Computerised Resources

Initially, we reacted against the idea of bringing humans into the same framework as the computer, because we felt this would reduce humans to the level of machines. However, BLOODCASE showed the need for viewing the two together. A comprehensive theory of organisational performance must also take other views than the mechanistic into account [12]. A mechanistic view is not harmful in itself; it is harmful used in isolation. This paper tries to develop a basic framework, which is needed prior to addressing more complex problems.

For many purposes, viewing humans and computers in the same way makes sense [10]. As an example of a similarity between human and computerised resources, both computers and humans may be multiprogrammed. Both can for example read and type at the same time.

Context switching is another example of a similarity. The context of a workflow is lost when a person starts another workflow, and it will take some time before the context of the old workflow is restored. For computers the overhead of context switching between processes, where working registers have to be reloaded, is well-known.

Both in organisations and in computer systems, a resource may move freely between several levels of abstraction, e.g. a human may type on the keyboard (low level), analyse the content of the screen and change it (higher), or assess the wider effects of following the routine required by the computer program (even higher level). A CPU in a computer will also work at several levels of abstraction, because it is used by for example both a DBMS and file system, the latter on a lower level of abstraction than the former.

10 Differences between Human and Computerised Resources

The most important difference between humans/organisations and computers is that humans cover all the other organisational views of Morgan outlined in [12], not only the mechanistic view. When our framework is used to compare humans and computers, it is therefore very important to in addition take the other organisational views into account. As an example, in the struggle for efficiency (a mechanistic concept concerned with "doing things right") it is easy to loose sight of effectiveness (where all the other views are also relevant: "doing the right things"). Improvements of organisational efficiency may for example lead to painful stress injuries, which is not very effective for the persons involved. As another example of unbalance between efficiency and effectiveness, increases in information system efficiency in an organisation may be taken out in terms of increased amount of information, improved quality of the information and better presentation, but not in increased effectiveness of the organisation, which BLOODCASE is an example of. Again, this leads to suboptimal solutions.

Humans are creative and take initiatives (make requirements) and can perform ad-hoc work (for satisfaction of needs which are hard to formalise). Hence, it is not possible to specify a human process exactly, in contrast to a computer process where this is possible [19]. There will often be a gap between the specification of a human process and the way it is actually performed. This gap should in the ideal case be non-existing for routine workflows. But even here the human ability to fix flaws is often vital as has been shown in BLOODCASE, where the personnel used the old manual routine to get acceptable performance during high workload. Adaptive behaviour is related to the concept of ad-hoc processes, which requires further study.

Another important difference is that the characteristic time elements between human and computerised resources differ with several orders of magnitude [4].

11 Conclusions and Further Work

Application of the framework to BLOODCASE was encouraging. The problem with the transfusion process described in Section 6 shows the tight coupling between a computerised information system and a manual information system. These systems do not use the same type of resource, and it is shown how the basic framework in this paper will solve the problem. The framework places human beings at the top of the hierarchy, in the sense of "actors" or initiative takers. However, the framework also treats humans in the same way as computers, as "resources" used to process, communicate, store and retrieve information.

Apart from the Gas Sales Telex Administration Case Study [1, 3], more complex examples of the interaction between humans and computers are needed to thoroughly evaluate this framework. More work is also needed in the following areas:

Integration with goals To improve a process, one must know something about the intentions as described by Yu [19]. Yu's thesis models goal and softgoal relationships in addition to resource and task relationships. It would be interesting to integrate these relationships into the PPP/SP framework.

Integration with task analysis Task analysis [14] is the study of how people actually do work with software on a low level of abstraction: i.e. at the level of key strokes in user interfaces.

References

[1] Gunnar Brataas. *Performance Engineering Method for Workflow Systems: An Integrated View of Human and Computerised Work Processes.* PhD thesis, Norwegian University of Science and Technology (NTNU), July 1996. Available from http://www.fou.telenor.no/brukere/gunnarb.

[2] Gunnar Brataas, Peter Hughes, and Arne Sølvberg. Integrated Management of Human and Computer Resources in Task Processing Organisations: A Conceptual View. In *27th Hawaii International Conference on Systems Science. Minitrack: Modelling the Dynamics of Information Systems and Organisations*, volume IV, pages 703 – 712. IEEE Press, 1994.

[3] Gunnar Brataas, Peter Hughes, and Arne Sølvberg. Performance Engineering of Human and Computerised Workflows. In Antoni Olivé and Joan Antoni Pastor, editors, *Advanced Information Systems Engineering – CAiSE*97*, number 1250 in LNCS, pages 187 – 202. Springer, Barcelona, Catalonia, Spain, June 1997.

[4] Peter J. Denning. Work Is a Closed-Loop Process. *American Scientist*, 80(4):314 – 317, July – August 1992.

[5] Remko C.J. Dur. *Business Reengineering in Information Intensive Organisations.* PhD thesis, Technical University of Delft, 2600 AJ Delft, The Netherlands, September 1992.

[6] Kenneth Fløstrand. Estimating Organization Work: A Blood Bank Case Study. Master's thesis, The Norwegian Institute of Technology, The University of Trondheim, Dec 1991.

[7] Jon Atle Gulla, Odd Ivar Lindland, and Geir Willumsen. PPP, An Integrated CASE Environment. In *Advanced Information Systems Engineering*, pages 194 – 221, Trondheim, Norway, May 1991. Springer-Verlag.

[8] Peter H. Hughes. SP principles. Technical report, STC Technology o59/ICL226/0, July 1988.

[9] Edward D. Lazowska, John Zahorjan, G. Scott Graham, and Kenneth C. Sevcik. *Quantitative System Performance - Computer System Analysis Using Queueing Network Models.* Prentice-Hall, Englewood Cliffs, New Jersey 07632, 1984.

[10] Thomas W. Malone. *Cognition, Computation, and Cooperation*, chapter Organizing Information Processing Systems: Parallels between Organizations and Computer Systems, pages 56 – 83. Ablex, Norwood, N.J., 1990.

[11] Ronni T. Marshak. Workflow White Paper. In *Workgroup Computing Report*, pages 15 – 42. Patricia Seybold Group, 1994. Volume 16, Number X.

[12] Gareth Morgan. *Images of Organizations.* Sage Pubications Inc., 1986.

[13] Andreas Lothe Opdahl. *Performance Engineering during Information System Development.* PhD thesis, The Norwegian Institute of Technology, The University of Trondheim, November 1992.

[14] Stephen J Payne. Task-Action Grammars: A Model of the Mental Representation of Task Languages. *Human-Computer Interaction*, 2:93 – 133, 1986.

[15] Wenche Sjøgren. Organizational Performance: A Blood Bank Case Study. Master's thesis, The Norwegian Institute of Technology, The University of Trondheim, December 1993.

[16] Connie Umland Smith. *Performance Engineering of Software Systems.* Addison-Wesley, 1990.

[17] Jari Veijalainen, Aarno Lehtola, and Olli Pihlajamaa. Research Issues in Workflow Systems. In *Proceedings of the 8th ERCIM Database Research Group*, Trondheim, NORWAY, August 1995.

[18] Vidar Vetland. *Measurment-based Composite Computational Work Modelling of Software.* PhD thesis, The Norwegian Institute of Technology, The University of Trondheim, August 1993.

[19] Eric Yu. *Modelling Strategic Relationships For Process Reengineering.* PhD thesis, University of Toronto, December 1994.