Customer: Kjell Lemström Instructor: Anita Kasari

OT4: pics99

Project Closing Document

Timo Immonen
Jussi Lindgren
Sampo Pyysalo

Helsinki May 27, 1999

Project Closing Document
Software Engineering Project
UNIVERSITY OF HELSINKI
Department of Computer Science

Contents

Version Number	Comments	Date/Author
0.1	document started	25.5.99 /TI
0.9	document in good shape	25.5.99 /TI
1.0	final changes+post mortem	27.5.99 /TI

1 Introduction

PICSearch is a development tool for content-based image retrieval algorithms. It is aimed to serve as a testing platform for new retrieval algorithms. It can also be used as a conventional image cataloging program.

The initial version PICSearch was developed at Helsinki University of Technology. First version, **PICSearch v1.0**[1], was restricted to use a local image database only. Further development at University of Helsinki resulted in **PICSearch v2.0**[2], a version enhanced with rudimentary network capabilities. A third version of program was also developed at University of Helsinki. **PICSearch v3.0**[3] added a cache implementation for greater network efficiency and also featured an improved graphical user interface.

The PICSearch program was further developed by pics99 software engineering project at University of Helsinki during spring 1999. This development resulted in **PICSearch v3.1**. Kjell Lemström (HY/TKTL) acted as project customer, and the project group was supervised by Anita Kasari (HY/TKTL). Project team members were students Timo Immonen (acted as project manager), Jussi Lindgren and Sampo Pyysalo.

2 Overview of the PICSearch

2.1 Structure

PICSearch program is structured in four distinct operational modules: The client software contains a graphical user interface. The host is the central point of the server side of the PICSearch. The server is an independent service provider process. One server is created by the host process per a connected client process. The database is a separate module which provides both servers and the host process with a database interface services (see Figure 1).

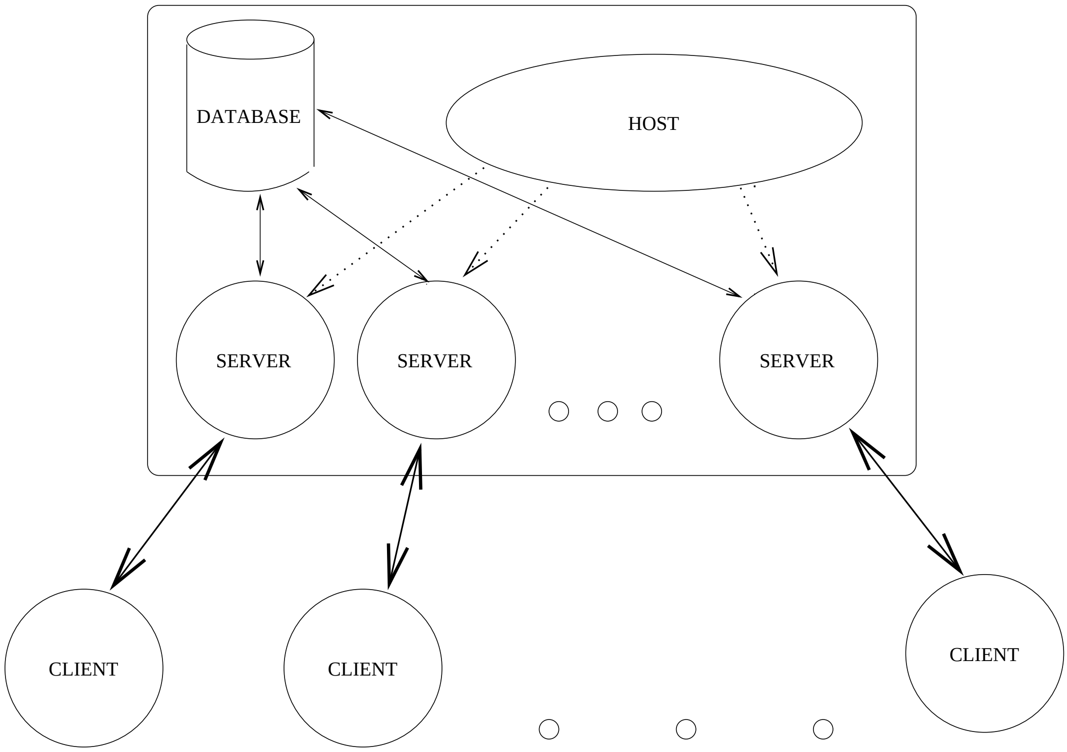

Figure 1: PICSearch functional modules (after client connection).

2.2 Operation

Normal operation of the PICSearch consists of a host process running on a server system. A user wishing to utilize PICSearch system starts a client front end of the PICSearch on his/her local, network capable workstation. Local connections with host/server and the client running all on a same system are also supported.

A client package is also used as a PICSearch system console for various maintenance operations, for example for adding new algorithm libraries and user management. In **PICSearch v3.1** all system configuration information is held in human-readable text files, and the system can also be administrated using a normal text editor.

On server side the host process monitors a well-known port address, and creates a socket connection when it receives client's request. It then creates a server process and assign it to the socket. Server process communicates with the connected client module and implements queries and other image database transactions. It also handles various system management activities when performed using the PICSearch program, such as adding a new search algorithm or a user account.

2.3 New Features

Major modifications and additions in **PICSearch v3.1** include a method for adding new search algorithm libraries to a running PICSearch server, a new database engine and a revised internal communication. Excluding the algorithm entry system, only very few user-observable changes have taken place. However, potential developers using existing **PICSearch v3.1** as a starting point for a proprietary image content based search systems will greatly benefit from code cleaning and organizing work also done during the pics99 project.

3 Project Life cycle

3.1 Timetable

The pics99 project was officially started January 28th, with revised project timetable (see attachment 1) being accepted February 1st. Due to the facts that no one of the pics99 staff had any prior knowledge of the PICSearch program, and that the pics99 project was fourth of the PICSearch projects, no firm checkpoints for the first half of the project were set. Instead the first half of the project was spent by streamlining code and removing unneeded legacy code. This also served to let the team gain familiarity with the code, and insights to possible new features and improvements.

First part of the project completed in time. As the aims of the project were clarified, some documentation specified in original timetable and student project guidelines was left out or merged to form a more coherent and applicable document for the needs of the pics99 project.

Second part of the project proceeded on time until a workload sharing problem became apparent. An untimely flu aggravated the problem, leading to several missed milestone deadlines, and to an eventual postponement of the project closing by more than two weeks from projected May 10th to May 27th. This includes using up a specifically reserved "panic week", so the effective over scheduling of manpower resources is only 10 days. For final timetable see attachment 2.

More serious is a reduced amount of testing allowed by forcefully revised timetable. However, even at the start of the project a series of reservations concerning complete testing and documenting of the PICSearch program was acknowledged by both the customer and the project controllers. Rigorous module testing has had to play substitute for complete integration and system testing. No acceptance testing has been performed by project close date.

3.2 Management

The pics99 project team met semi-regularly on mondays and thursdays. A meeting was always recorded in minutes in finnish. Regular items discussed were the project situation, workload and task assignments. Various milestone inspections were made, though only one inspection was "official". Other items were discussed when necessary.

Main forum of communication was email. Due to the differing personal timetables and working environment preferences team met only once or twice for actual coding and documenting purposes. Also the code related parts of the project were highly modular, and as such suitable for individual work efforts.

Documentation tasks were mainly divided on bases of actual implementation task assignments. To assist on project management and to aid the implementation following documents were written: Project Plan (including timetable chart), Design Requirements, Design Specification and Implementation of changes and Testing and Implementation Document.

4 Project Time Usage

Time usage during the pics99 project was followed by daily personal work logs. A weekly report was generated and sent to the project instructor. While the personal logs were kept using same format, no formal policy of dividing various task to various log categories was maintained. This most likely resulted in deviations on observed categorized time usage. However, usually the team members had several open tasks at hand, so different activities of different team members need not necessarily be a logging uncertainty. On attachment 3 several summary graphs of time usage during pics99 project are presented.

5 Post Mortem Analysis

This section is semi-formal analysis minute of pics99 project group post mortem meeting, held 27th May.

5.1 What Went Wrong?

1. Dividing workload to individual tasks: Project manager did not insist on cross-checking time estimates presented by individuals, therefore some task were much more demanding time-wise than other tasks.

2. Personnel risk: When one team member suffered an untimely flu attack, no replacement or workload re-sharing was done. However, no personnel reserves are usually available on student course projects, so personnel risks can not be effectively anticipated. Also, the knowledge gap of one team member did not allow totally dynamic workload sharing.

3. Overall workload: The team agrees that less could have been done better, but also that in this particular case less done well would still be in effect worse than more done worse is.

4. Old code did fight back: The overall burden presented by old legacy code and unclear modules was thought to be less than it turned out to be. By cutting out and completely re-writing larger parts of the code earlier the whole project could have been substantially speeded up. In addition, the result would have been much clearer than even present "polished" version is.

5.2 What Went Right?

1. Team spirit was good, even when abilities of team members to take part in project were somewhat unequal.

2. Communication: While first viewed skeptically by supervisor, email communication was more than adequate to facilitate full information sharing. Project situation and work task assignments were generally made during official meetings, and hard details of design and implementation ironed out with email communication.

3. Overall work methods and project roles: While workload sharing in itself was slightly amiss, the methods used during the project were perfectly suited to circumstances and personal tastes and timetables. As the first part of the project (code removals etc.) was in any case doable only by rigorous individual efforts, and the second part (new db, alg. libraries) was easy to divide in coherent subtasks, individual efforts would not have gained much synergy (indeed, social interaction might have slowed down implementation) by enforced team work in same physical location.

5.3 What We Have Learned

1. Tools: Some of us have gained familiarity with new tools, some have freshened up their previous experience.

2. Focus: Most important aspect of our project experience is clarification of the need to concentrate to the essential. During a project no secondary tasks should cloud the essential facts of the work to be done.

3. Responsibilities: Hardest to catch implementation errors seem to be those that really are not part of anybody's specific problem space. An individual totally free of actual coding work, but knowledgeable of the tools employed in project would be a valuable asset in coordination and work coverage checking.

5.4 Course Critique

The team wishes to take this opportunity to offer some constructive critique about student software engineering project courses in general.

Not all documentation is always applicable: In some projects like pics99 much of the documentation required about the course project is not essential to the project as a whole. On the other hand, some different documentation might be needed at different times. By forcing the team to wait for implementation phase and by requiring certain documents on certain dates, time better used on actual coding was wasted.

We do not question need for documentation in general, only the applicability of formal guidelines and document timetables designed to assist a team producing a totally new program to a project continuing to develop an existing software. However, designing a documentation structure for a continuation project would be a project in itself.

A sterner advice against over-ambitious undertakings by instructors might also result in more finished project product, but then, a hectic timetable with only marginally usable product software is a real life experience and a business wide industry standard. =)

References

[1]
K. Lemström, J. Korte, P. Kuusi, P. Kyheröinen and P. Päiväkumpu:
PICSearch - A platform for Image Content-based Searching Algorithms, (Technical Report C-1997-48), proceedings of the Sixth International Conference in Central Europe on Computer Graphics and Visualisation 98 (WSCG'98), edited by Vaclav Skala, Chzech Republic, February 9–13, 1998, pp. 222–229.
`http://www.cs.helsinki.fi/ klemstro/publications.html`

[2]
P. Kuusi, K. Lemström, P. Kanerva, S. Koponen, S. Männynsalo, J. Riihimäki, J. Teinilä and T. Virtanen:
A Client-Server Tool for Image Retrieval Applications, In: Proceedings of the HeCSE Workshop on Emerging Technologies in Distributed Systems, Helsinki University of Technology, Department of Computer Science and Engineering, pp. 45–46 (Research Report A50). Otaniemi, Finland. March 1998.

[3]
S. Kaski, T. Koskinen, J. Leinonen and M. Pennannen:
pics98, Implementation Specification, Helsinki, Finland, September 10, 1998.
`http://www.cs.helsinki.fi/group/pics98/`