Global Software Development: Issues, Solutions, Challenges

Parastoo Mohagheghi
Dept. Computer and Information Science (IDI)
University of Science and Technology (NTNU)
Trondheim, Norway
parastoo@idi.ntnu.no

Trial lecture, 21 September 2004

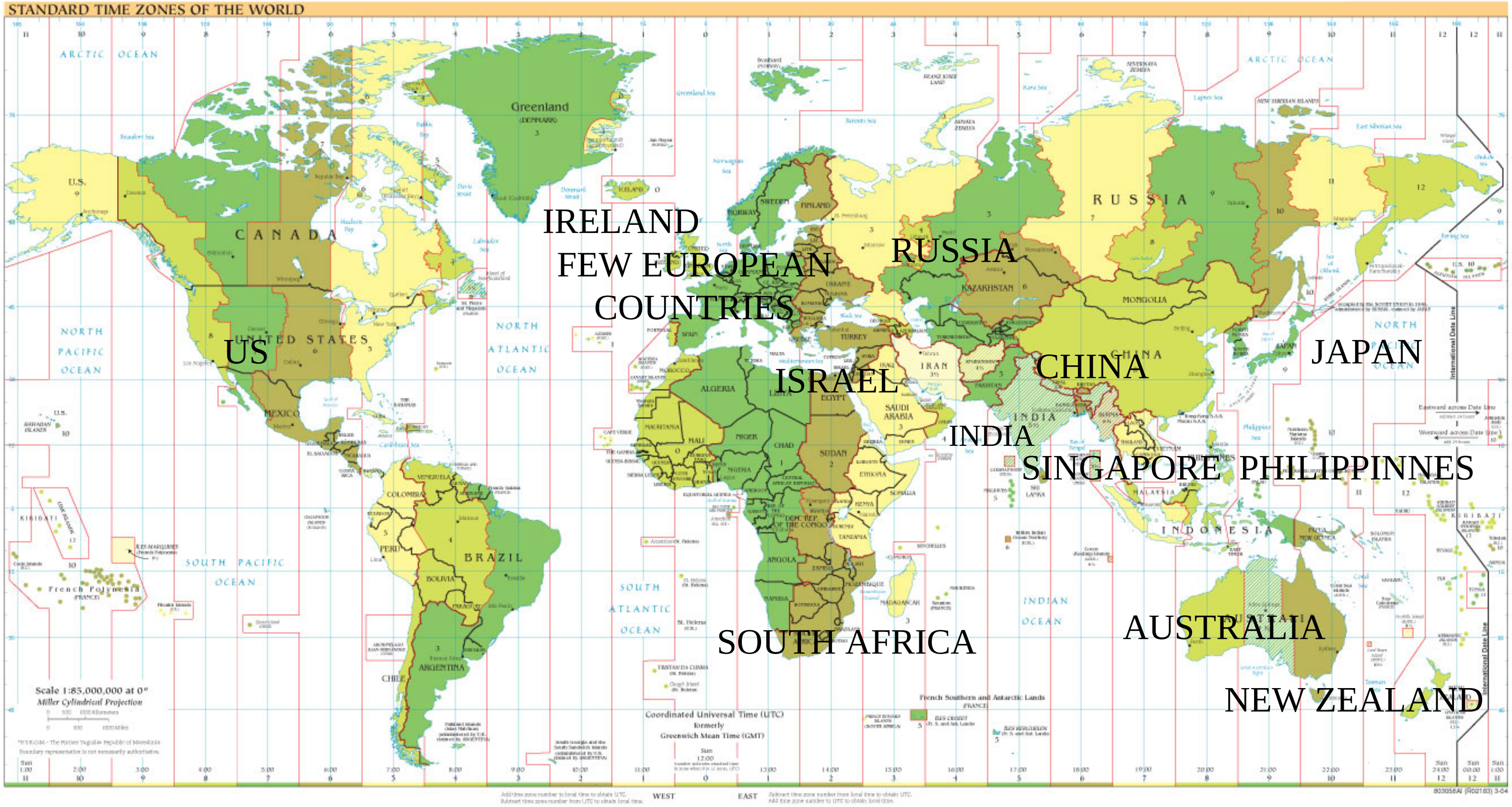
What is Global Software Development (GSD)?

- Sahay (UiO) defines it as “software work undertaken at geographically separated locations across national boundaries in a coordinated fashion involving real time (synchronous) and asynchronous interaction” .
  - Involves ***communication*** for information exchange.
  - Involves ***coordination*** of groups, activities and artifacts so they contribute to the overall objective.
  - Involves ***control*** of groups (adhering to goals and policies) and artifacts (quality, visibility & management).

Some figures on the extent of GSD

- 40% of the Fortune 500 companies use GSD, and 185 of these outsourced to India alone [Global Business Technology, NASSCOM, 2000].
- Upwards to 50 nations are participating in GSD [Carmel, 2001].
- IBM, British Airways, Alcatel, British Telecom and General Electric have moved parts of their software development to countries like Ireland and India [Khan, 2003].
- 80% of the Irish software industry’s output is exported [Cochran, 2001].
- Gartner Dataquest has projected that IT outsourcing will reach $159 billion by 2005 [Laplante, 2004].
- 87 000 open source projects are hosted at SourceForge.net on Sept. 2004.

Worldwide development centers



IRELAND

FEW EUROPEAN COUNTRIES

RUSSIA

US

ISRAEL

CHINA

JAPAN

INDIA

SINGAPORE

PHILIPPINNES

SOUTH AFRICA

AUSTRALIA

NEW ZEALAND

Why Global Software Development? The “most given” answers

Solving local IT skills shortage,

Motorala in 2000 had only 20% of required staff for 3G trial.

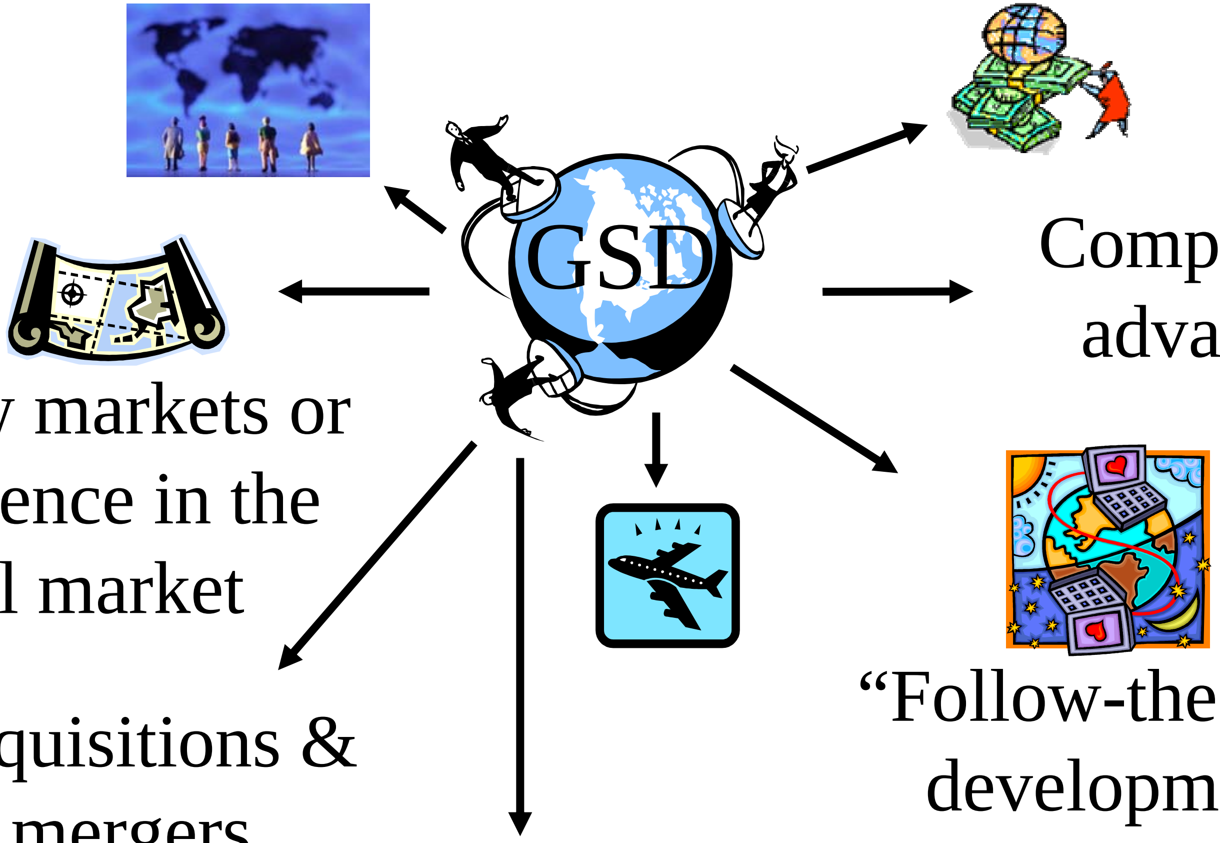
Cost saving,

overseas projects cost about $25 per hour vs. US $75 per hour

Remain focused on core competencies,

(outsourced functions are complex, important or distasteful [Bruce Schneier, security expert]

Why Global Software Development?
A more complete picture



GSD

New markets or presence in the local market

Competitive advantage

Acquisitions & mergers

“Follow-the-sun” development

Improved quality?
Impact on policies?

The four approaches to GSD

1. Intra-organizational or legally related companies;
   - E.g. Siemens, Lucent Technologies, IBM, Ericsson
2. Inter-organizational or outsourcing ;
   - IT infrastructure, data centers, embedded software, maintenance or even software applications.
3. Open Source Software or non-organizational;
   - OS like Linux, programs for programmers (editors, compiler).
4. Services or components over Web;
   - Application Service Providers (ASP), pay-per-use services and recent component markets.

Benefits and risks of intra-organizational or inter-organizational global development

Benefits	Risks
Solution to IT skills shortage	Threat of opportunism, security and trust concerns, training, cultural issues
Cost efficiency	Hidden or unexpected costs, delay, the value and cost is intangible and long-term oriented, detailed spec.
Competitive advantage	Loss of control
Follow-the-sun development or Round-the-clock service	Geopolitical risks, coordination problems
New markets	Legal issues

Issues in GSD

- ***Strategic issues:*** when, to whom and how, task allocation.
- ***Communication issues:*** distance, time zone difference, infrastructure support, distinct backgrounds, lack of informal communication.
- ***Coordination*** complexity
- ***Cultural issues:*** power distance, individualism vs. collectivism, attitude to time etc.
- ***Geographical dispersion:*** vendor support, access to experts, software practices that need face-to-face comm.
- ***Technical issues:*** information and artifact sharing, software architecture.
- ***Knowledge management:*** slow communication, poor documentation, tacit knowledge, repositories etc.

Overview of the remainder

Challenges in Inter/intra org. GSD

Communication

Culture

Organizational models

Work allocation

Open Source answers

Literature overview

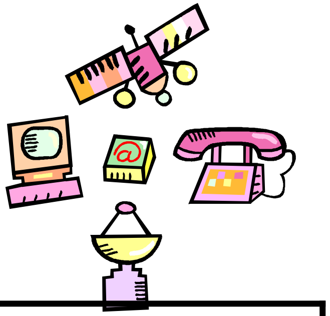
Summary

Ericsson example

Communication

- Communication type:
  - ***Formal:*** for routine coordination, formal specifications or inspections or meetings.
  - ***Informal:*** in the face of changes, informally captured requirements or out-of-date documents.
- Tom Allen, 1977: The frequency of communication among engineers decreased with distance. Even 30 meters distance (same building or floor) is like miles away.
- Some activities (e.g. requirement eng.) depend more on communication than others (e.g. testing).
- Communication patterns: more with local and less with remote people.

Communication means



<table>
<tr><th>Synchronous</th><th>Asynchronous</th></tr>
<tr><td>Phone, video conference, Net meeting, E-chat, Instant Messaging</td><td>E-mail, voice-mail, discussion list, on-line calendar</td></tr>
<tr><th colspan="2">Synchronous & Asynchronous</th></tr>
<tr><td colspan="2">Document sharing, Distributed Configuration Management (CM) systems, file transfer, remote access
Special tools such as distributed blackboards, intelligent CM systems, experience browser</td></tr>
</table>

Awareness is knowing what is going on. Awareness Systems filter information.

Cultural differences

- Edward T. Hall (1976) identifies two dimensions:
  - High context vs. low context cultures,
  - Poly-chronic vs. mono-chronic cultures.
- Geert Hofstede (2002) identified five views:
  - Power distance, collectivism vs. individualism, feminity vs. masculinity, uncertainty avoidance, long-term vs. short term orientation
- Other factors added by others:
  - Emotional vs. neutral, attitude to time, race, class, religion, attitude to governments and specific vs. diffuse.
- Cultures must be understood and respected. They could not be easily changed!

What is the impact of cultural differences on software development?

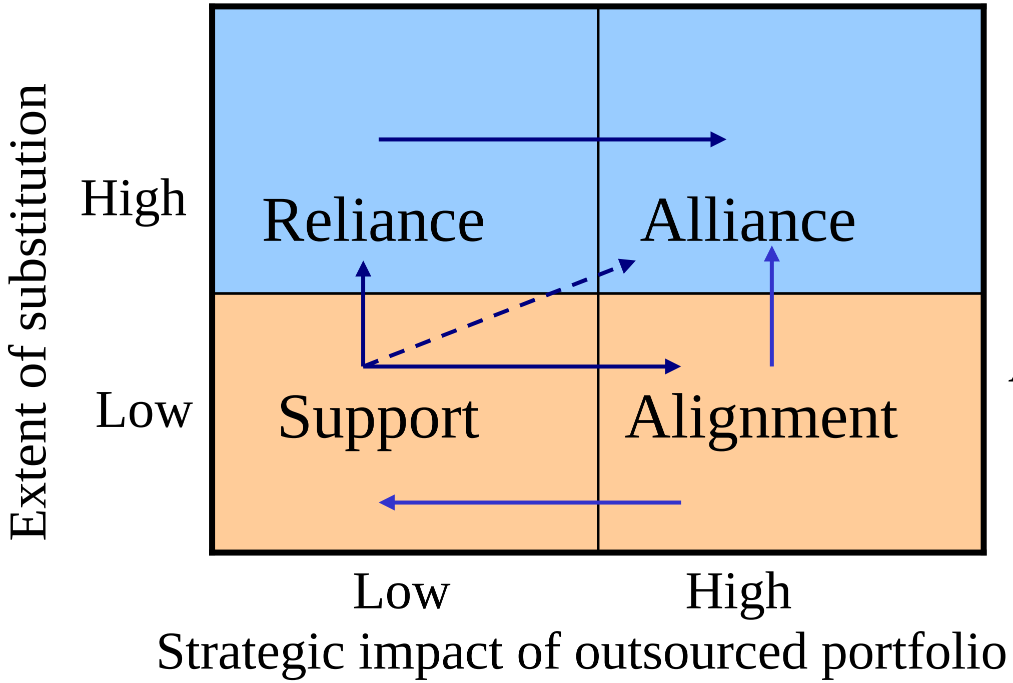
- *High power distance* -> hierarchical forms of communication and slow decision making (e.g. during requirement eng. in a study on Thai culture, 2000).
- *Uncertainty avoidance* -> waterfall development models and restrictive change process (a study in 2003 comparing Japan, US and India).
- *Collectivism* -> helping each other and correcting other people’s bugs (the same study, lowest for US).
- US clients normally work with extensive written agreements and frequent e-mail contact, while Japanese prefer more verbal and continuous communication and less (but more formal) e-mail contact [Krishna from Indian companies, CACM, April 2004].

Managing cross-cultural relationships

- Minimize cross-cultural issues or reduce intensive collaboration [Carmel 2001, Krishna 2004]:
  - Project types: Embedded software, OS
  - Contract types: contract programming, full ownership

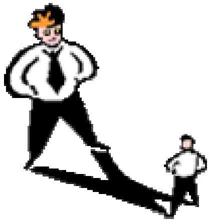
- Reduce cultural distance:
  - Locals on-site (75/25 rule), staff who bridge cultures, common processes and work environment, personnel exchange, culture liaison etc.
- Reduce temporal distance by communication means.
- Recognize limits and learn from the other part.
  - Reflect and share knowledge.

Relationships in IT outsourcing- FORT or Four Outsourcing Relation Types



Reliance: cost reduction, outcome-based, formal procedures to monitor vendor.

Alliance: strategic partnership, common objectives and informal comm. channels of importance.



Support: lowest set-up cost, out-come based, multiple bidding process.

Alignment: high-impact IS services, gain expertise, should integrate the new system with existing ones and provide in-house personnel for this, mostly dynamic and high-risk projects.

Work allocation

🟧 In-house

🟦 Outsourced

Alt.1: *Transfer by development stage*

Classic contract model, formal requirements, non-critical parts

🟧 Product Mngt.	🟧 RE	🟦 Design	🟦 Code	🟦 Unit test	🟦 System test	🟦 Maint enance

Implementation model, critical parts

🟧 Product Mngt.	🟧 RE	🟧 Design	🟦 Code	🟦 Unit test	🟧 System test	🟧 Maint enance

Product mngt. model, non-critical parts with further development

🟧🟦 Product Mngt.	🟦 RE	🟦 Design	🟦 Code	🟦 Unit test	🟦 System test	🟦 Maint enance

Maintenance model

🟧 Product Mngt.	🟧 RE	🟧 Design	🟧 Code	🟧 Unit test	🟧 System test	🟦 Maint enance

Mockus, IEEE Software 2001,Nissan, ICSE workshop 2004

Work allocation- *cont.*

Alt.2: *Transfer by functionality*

Melvin Conway: a (software) product’s structure reflects the organizational structure of the company that produced it.
David Parnas: software modularity should reflect the division of labor.
Solutions: systems or subsystems, horizontal layers or chunks of related functionality (Mockus). Requires independent units of development.

Alt.3: *Transfer by localization*

Alt.4: *Product line approach*

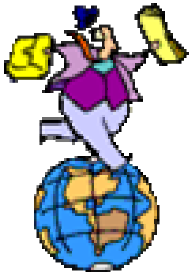
Independent architectural units that do not need customization are developed by collocated teams. Development of big chunks that need customization is replicated in several locations. (Elbert 2001 on Alcatel, Ericsson).

Common for all: Incremental approach

to facilitate communication and highlight ambiguities.

Other challenges in intra-org. and inter-org. global development

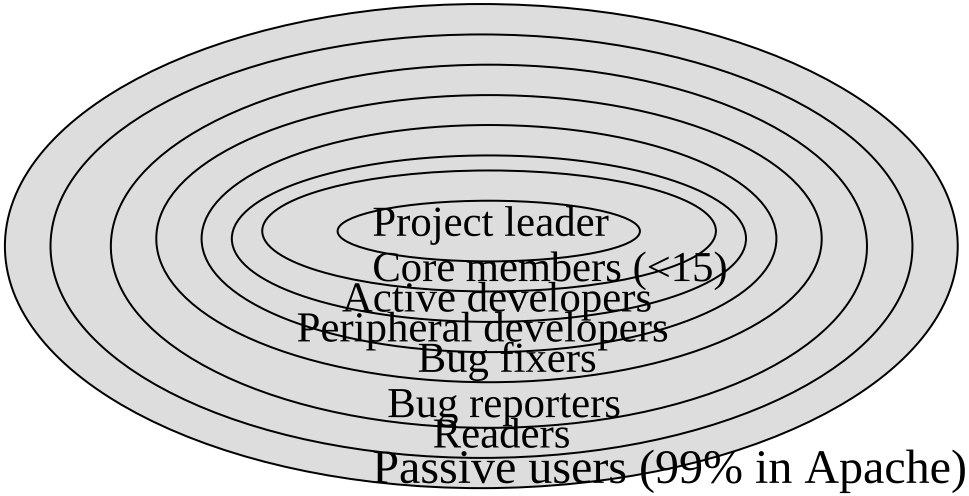
- Organizational models: local managers at each site (with common visions).
- Risk management, both at organizational level and project level.
- New roles (e.g. management) and changed roles (interfacing vendor).
- Pricing:
  - Fixed or dynamic tied to effectiveness measures for low-risk projects,
  - Time and material for high-risk projects,
  - Maintenance in-house or external.
- Adapting software processes.
- Tools.



Open Source Software

- Systems that give users free access to and the right to modify their source code.
- Usually solve all the problems of distributed development using only very simple communication tools such as email and newsgroups, and change management systems such as CVS or Bugzilla.
- Global access to developers. For example, the Apache project concerned with developing a HTTP (web) server included volunteers located in the US, Britain, Canada, Germany and Italy [Østerlie].
- Some characteristics:
  - There is no explicit system- level design or even detailed design.
  - There is no project plan, schedule or list of deliverables.
  - OSS projects have more frequent releases and rapid cycle time.

Open Source Software- *cont.*



General structure of an OSS community

No training,
Ad hoc alliances,
Free participation,
Work is not assigned,
Main motivation is learning and creation,
Status by the practice of using.

-The developers must be users; there is generally no requirements gathering.

-Bug finding and fixing is done by users -> low post-release defect density and high productivity due to independence of tasks.

- High modularized software, a highly capable (small) core team and informal coordination style -> speed

- Other processes and practices: voting, wedging and forking.

- Products: OS, compilers, system programs (Glass on Cathedral or Bazaar).

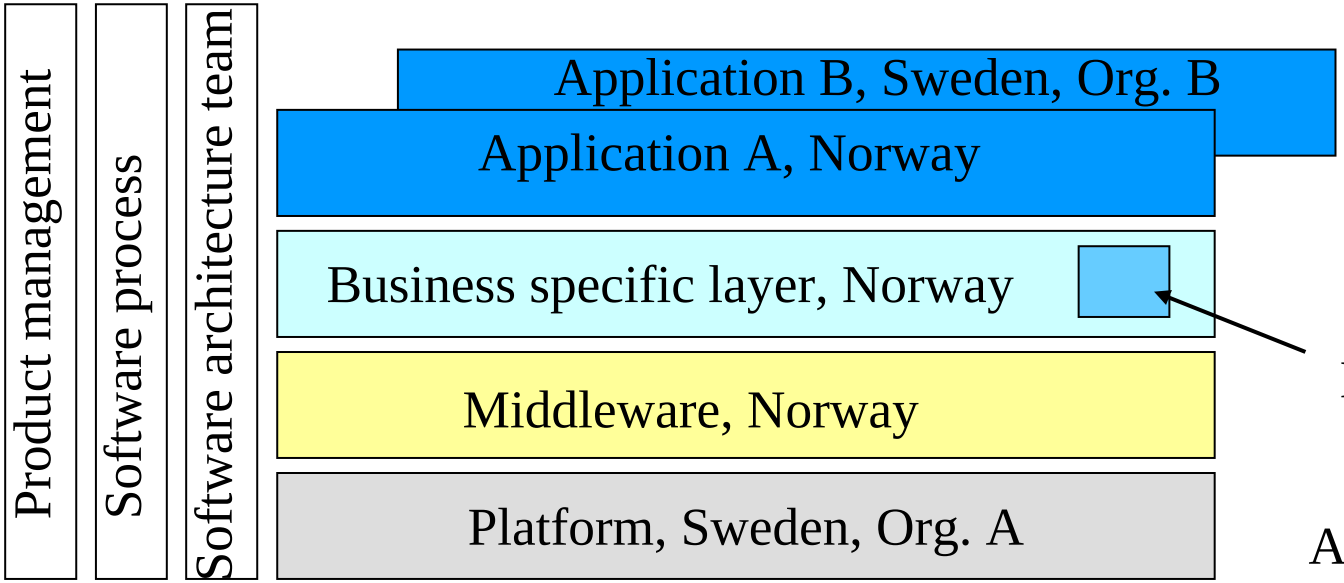
Quality impacts

- Only a few case studies report concrete impacts of GSD on product or project attributes:
  - Delay is the additional time it takes to resolve an issue when more than one site is involved. A study of modification requests (MRs) by at Lucent Technologies showed that single-site MRS took in average 5 days to complete, in contrast to 12,7 days for multi-site MRs (not related to size or number of changed modules, but to number of peoples involved) [Herbsleb et al. 2001].
  - Boland et al. report reduced productivity due to asynchronous communication (how much?) [ICSE workshop, 2004].
  - Others mention reduced defect-density and high productivity in OSS projects.
  - Collocated teams achieved an efficiency improvement during initial validation activities of over 50 percent [Ebert, 2001]
- Lack of quantitative results in general. What about cost, time-to-market, productivity in LOC etc.?

Empirical work

- Several descriptive case studies and results of interviews. Some formulative papers. A few papers on new tools, one quantitative study on MRs and one simulation of delay.
- on the supplier side, several papers from Indian researchers, but not from other nations.
- Research topics:
  - What type of projects should be (or might be) developed globally?
  - Who should work with whom? It should be answered together with management and social scientists.
  - What type of activities should be done collocated or distributed? For example should we do requirement eng. distributed?
  - Is the distribution cost justifiable?
  - What GSD needs in terms of practices, processes and tools?
  - Developing tools, methods and techniques.
  - Estimation, assessment of cost and impact on quality

Ericsson experience (from my thesis)



Product management

Software process

Software architecture team

Application B, Sweden, Org. B

Application A, Norway

Business specific layer, Norway

Middleware, Norway

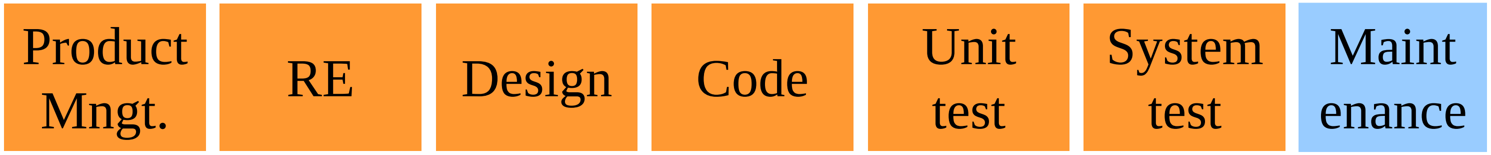
Platform, Sweden, Org. A

External,
Norway

Also extensive
use of consultants

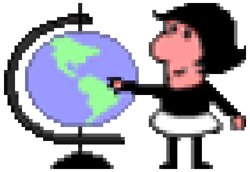
Initially:
Each organization did necessary requirement eng., design, coding and testing.
Final integration testing for the product was done in Germany.

Since 2002: maintenance of earlier releases is outsourced.



Product Mngt. | RE | Design | Code | Unit test | System test | Maint enance

Summary



- GSD takes several forms.
- Distance (time and space) creates many challenges in communication, coordination, organization, project planning and follow up, and work allocation.
- Advances in communication technology and tools have brought GSD in focus. Questions regarding benefits and risks need detailed analysis in each case. Research should also map approaches or methodologies to problem domains.

Sources

- ICSE workshops on OSS and GSD in 2001, 2002 and 2003.
- IEEE Software March/April 2001.
- Almost every issue of CACM since 2002!
- Annual Hawaii International Conference on System Sciences (HICSS) since 1997.
- ICSE main conferences.
- A few papers from other sources: METRICS'01, Organizational Science Journal etc.