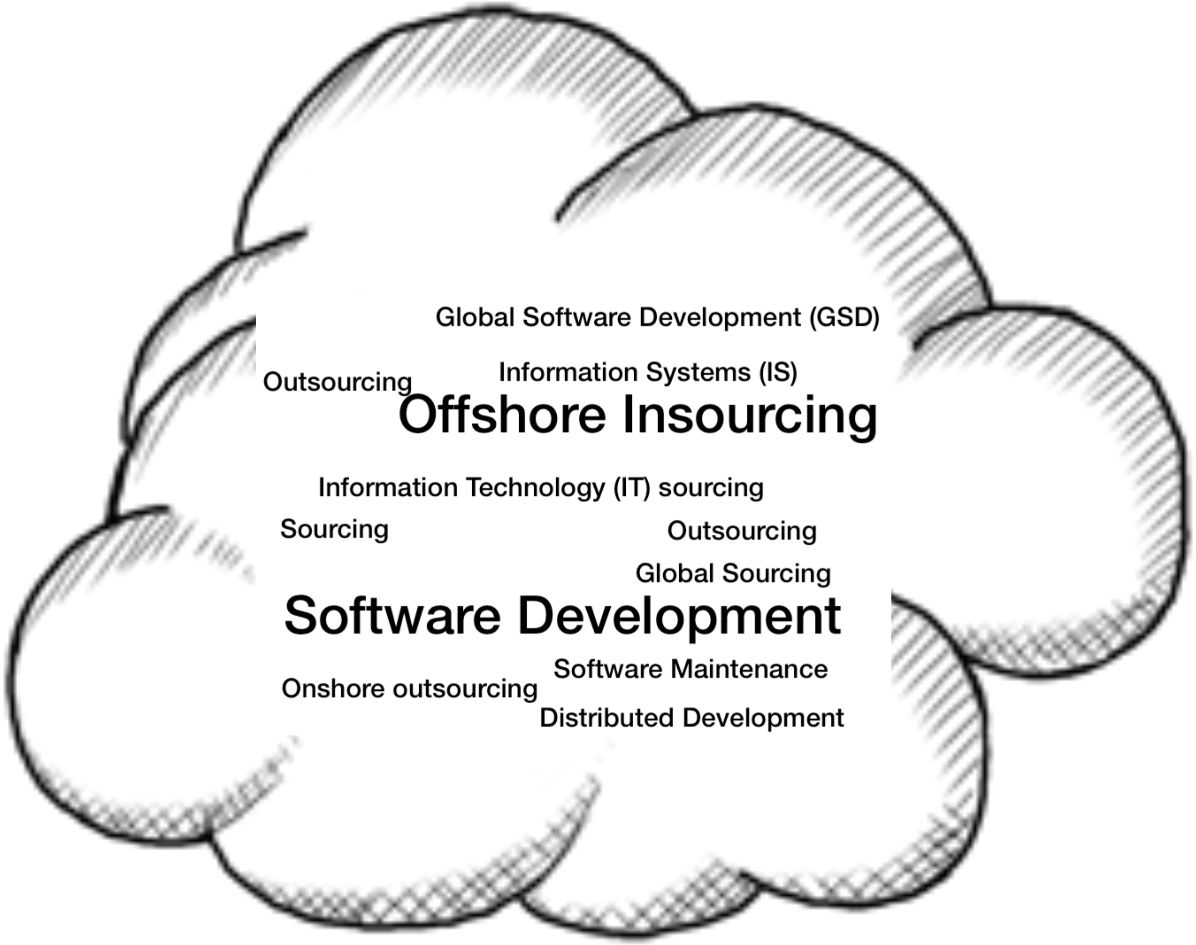
Global Software Engineering terminology and scenarios

There is no single type of global software projects

GSE is a complex phenomenon that unites many different collaboration modes and project organization models



Global Software Development (GSD)

Outsourcing

Information Systems (IS)

Offshore Insourcing

Information Technology (IT) sourcing

Sourcing

Outsourcing

Global Sourcing

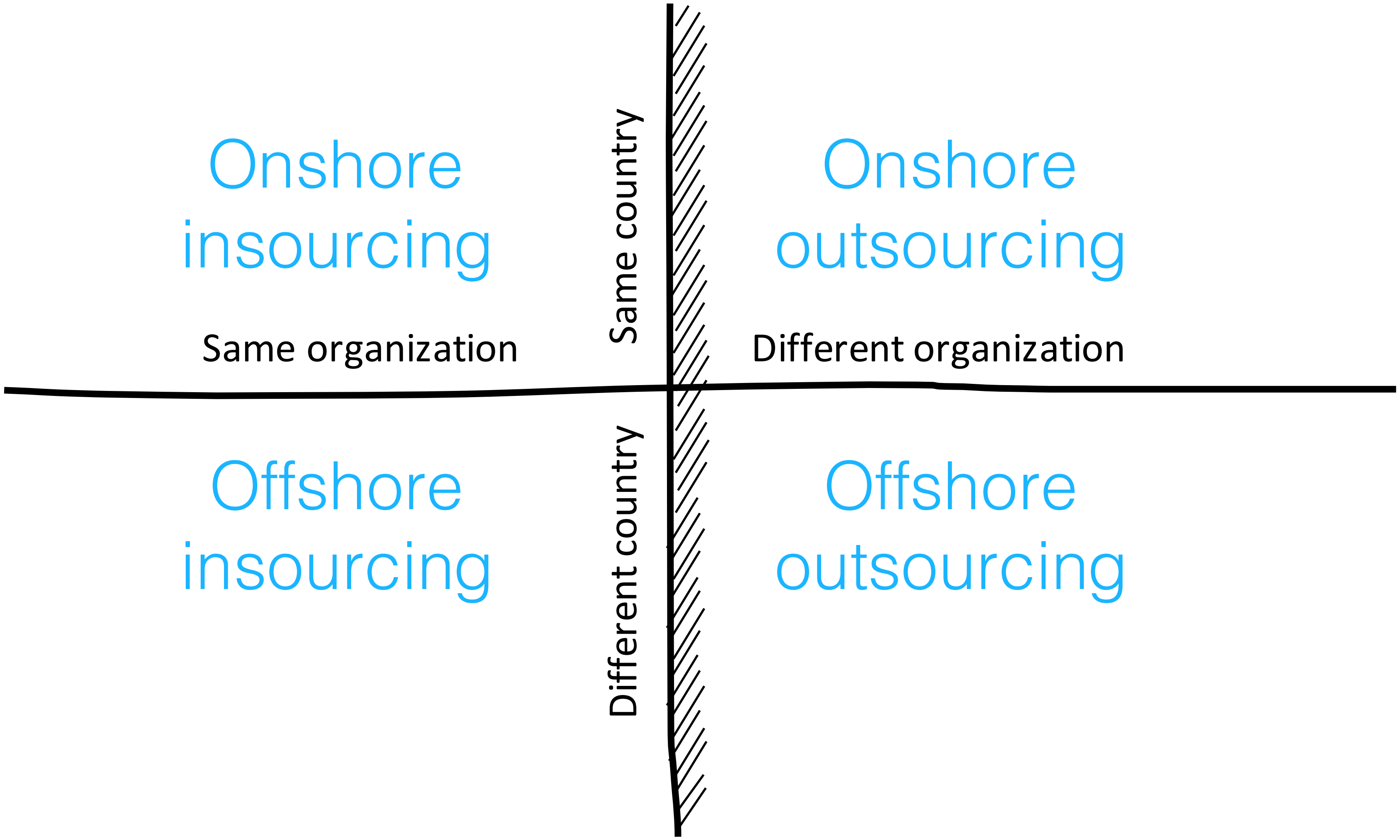
Software Development

Software Maintenance

Onshore outsourcing

Distributed Development

Terminology



Onshore insourcing

Onshore outsourcing

Offshore insourcing

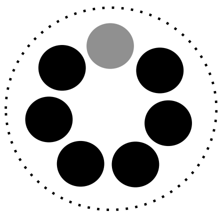
Offshore outsourcing

Same organization

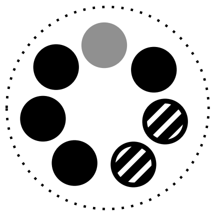
Different organization

Same country

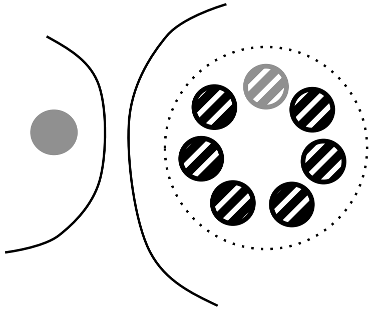
Different country



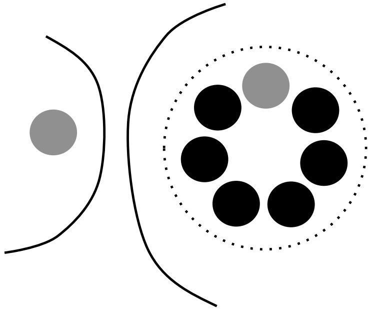
A) Traditional co-located team



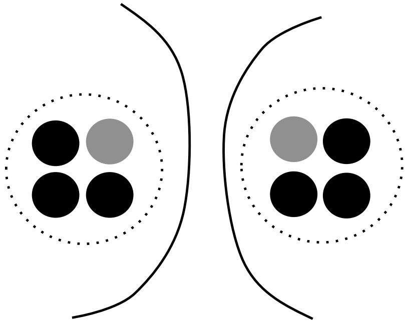
B) Co-located team with onsite consultants



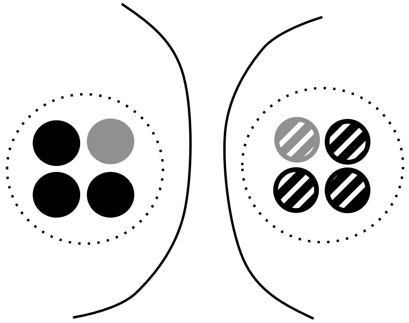
C) Non-distributed outsourcing project



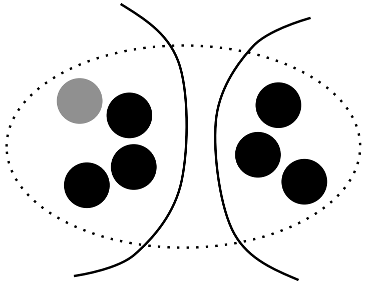
D) Non-distributed insourcing project



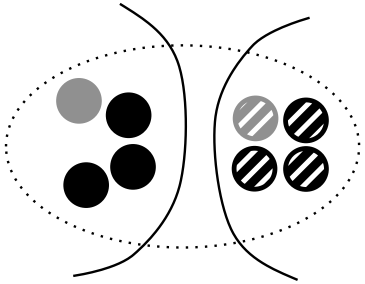
E) Distributed insourcing project with two distributed teams



F) Distributed outsourcing project with two distributed teams



G) Distributed insourcing project with one dispersed team



H) Distributed outsourcing project with one dispersed team

) *Location boundaries*

● *Project manager/Team lead*

▨ *Project manager/Team lead from a third party vendor*

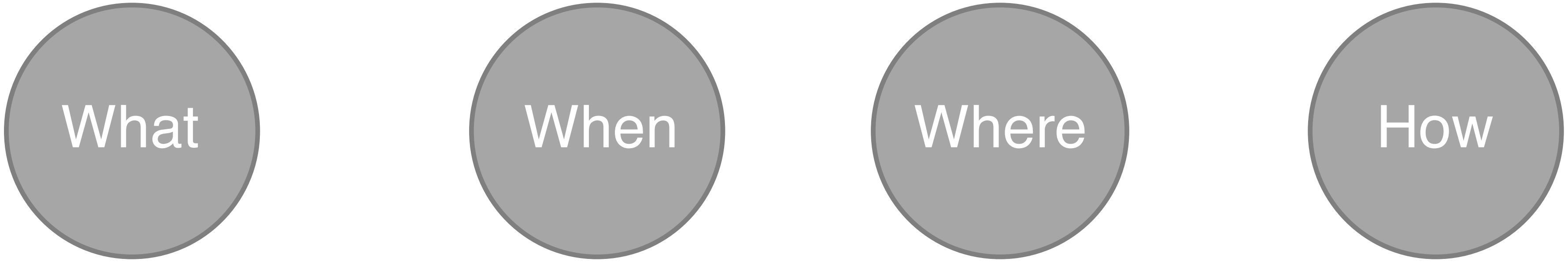
◌ *Team boundaries*

● *Project team member*

▨ *Team member from a third party vendor*

Not an exhaustive list of arrangements!

Each combination is unique



What

Type of work

When

Software life cycle

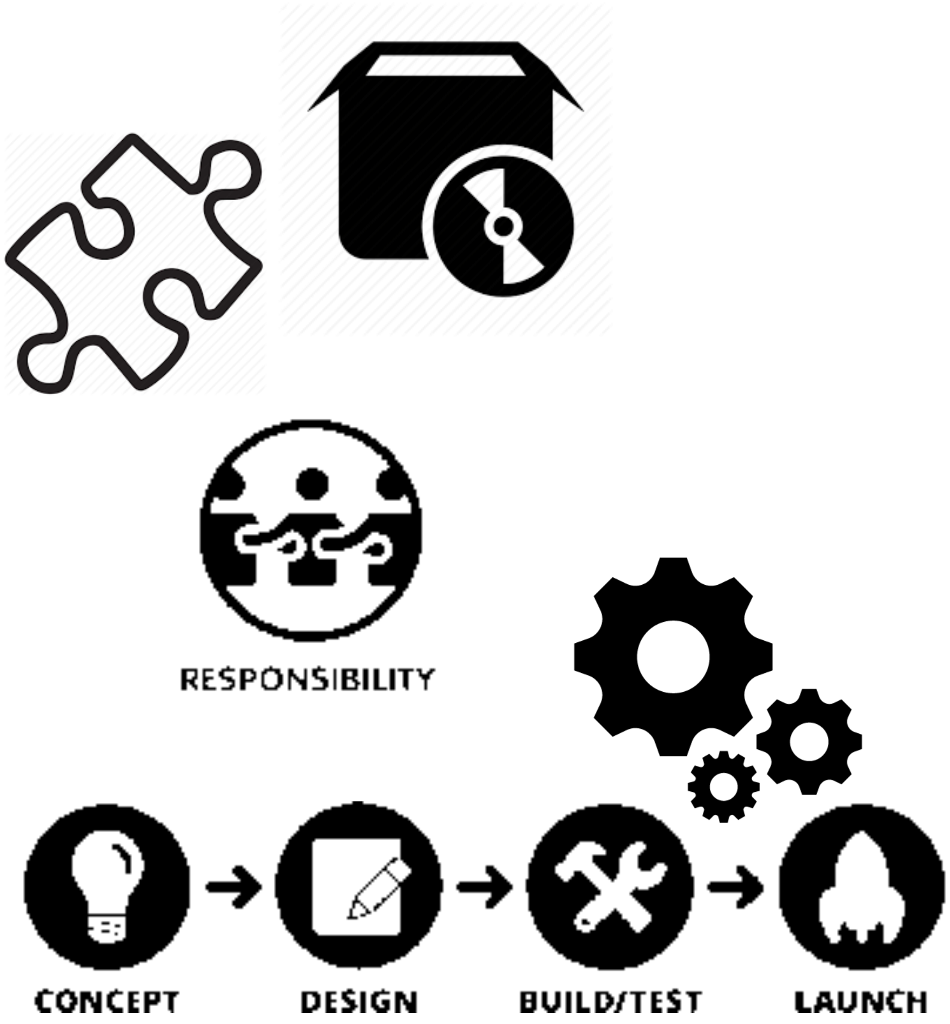
Where

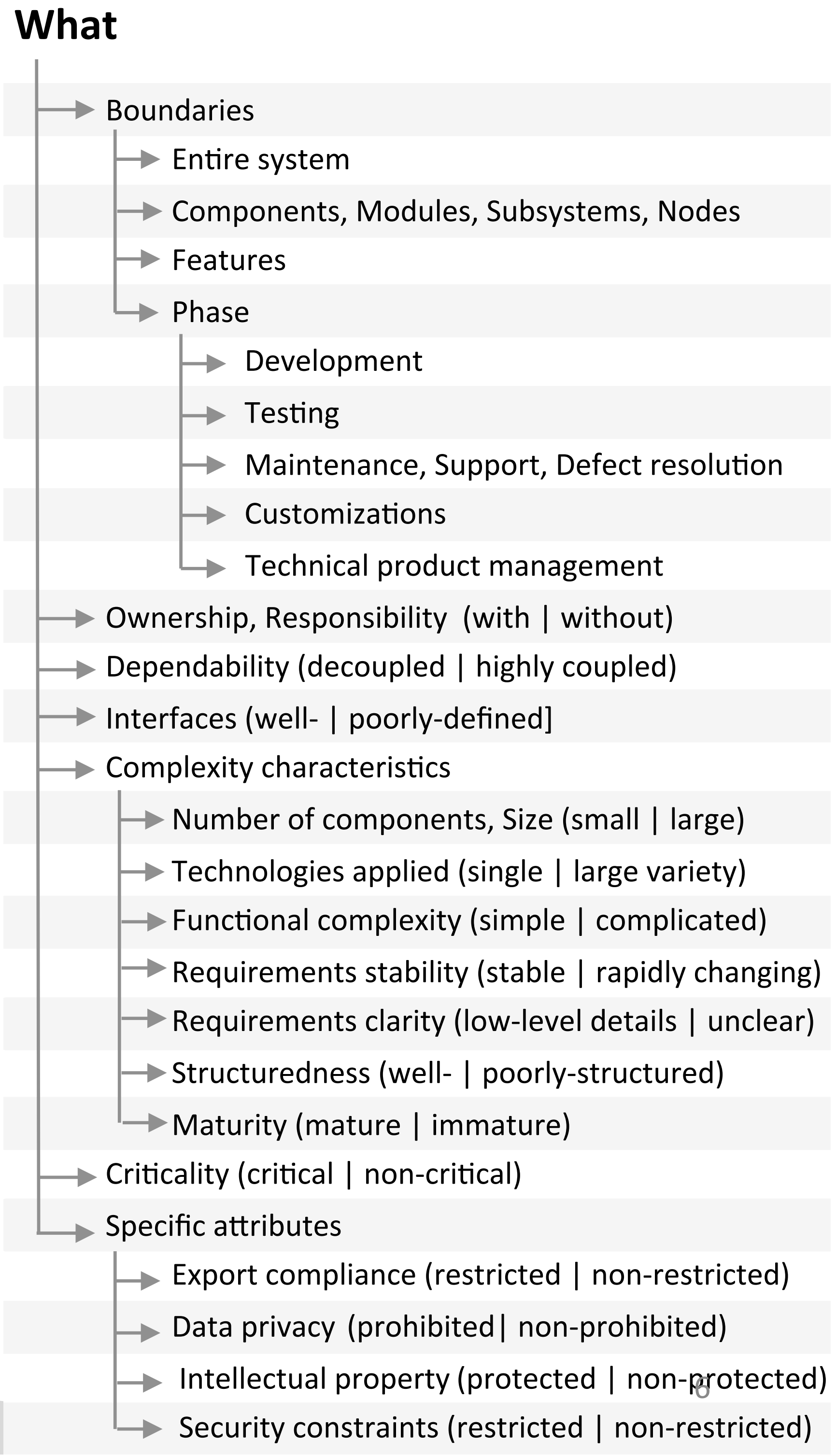
Destination

How

Type of collaboration

What is sourced?



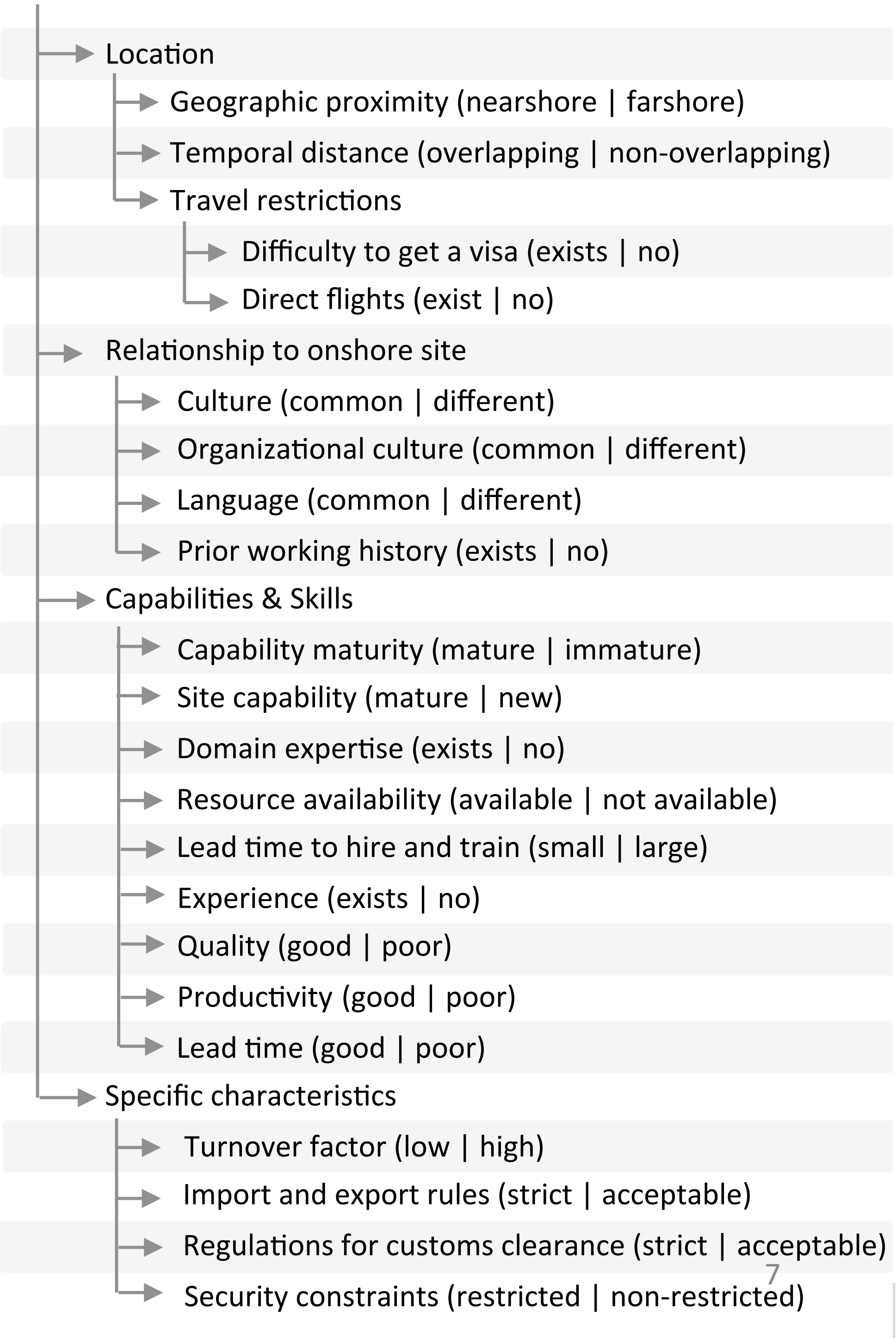


What

- Boundaries
  - Entire system
  - Components, Modules, Subsystems, Nodes
  - Features
  - Phase
    - Development
    - Testing
    - Maintenance, Support, Defect resolution
    - Customizations
    - Technical product management
- Ownership, Responsibility (with | without)
- Dependability (decoupled | highly coupled)
- Interfaces (well- | poorly-defined]
- Complexity characteristics
  - Number of components, Size (small | large)
  - Technologies applied (single | large variety)
  - Functional complexity (simple | complicated)
  - Requirements stability (stable | rapidly changing)
  - Requirements clarity (low-level details | unclear)
  - Structuredness (well- | poorly-structured)
  - Maturity (mature | immature)
- Criticality (critical | non-critical)
- Specific attributes
  - Export compliance (restricted | non-restricted)
  - Data privacy (prohibited| non-prohibited)
  - Intellectual property (protected | non-protected)
  - Security constraints (restricted | non-restricted)

Where is sourced?

Where

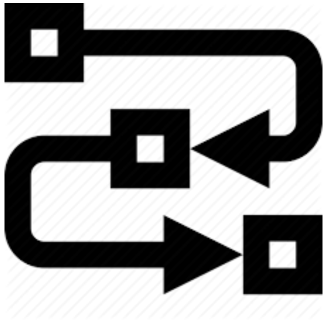


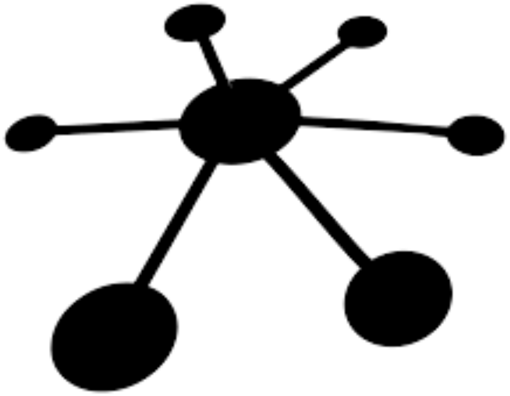
- Location
  - Geographic proximity (nearshore | farshore)
  - Temporal distance (overlapping | non-overlapping)
  - Travel restrictions
    - Difficulty to get a visa (exists | no)
    - Direct flights (exist | no)
- Relationship to onshore site
  - Culture (common | different)
  - Organizational culture (common | different)
  - Language (common | different)
  - Prior working history (exists | no)
- Capabilities & Skills
  - Capability maturity (mature | immature)
  - Site capability (mature | new)
  - Domain expertise (exists | no)
  - Resource availability (available | not available)
  - Lead time to hire and train (small | large)
  - Experience (exists | no)
  - Quality (good | poor)
  - Productivity (good | poor)
  - Lead time (good | poor)
- Specific characteristics
  - Turnover factor (low | high)
  - Import and export rules (strict | acceptable)
  - Regulations for customs clearance (strict | acceptable)
  - Security constraints (restricted | non-restricted)

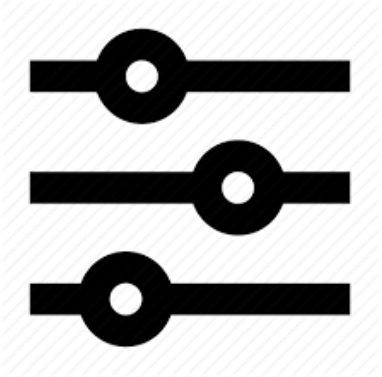
How is sourced?

How

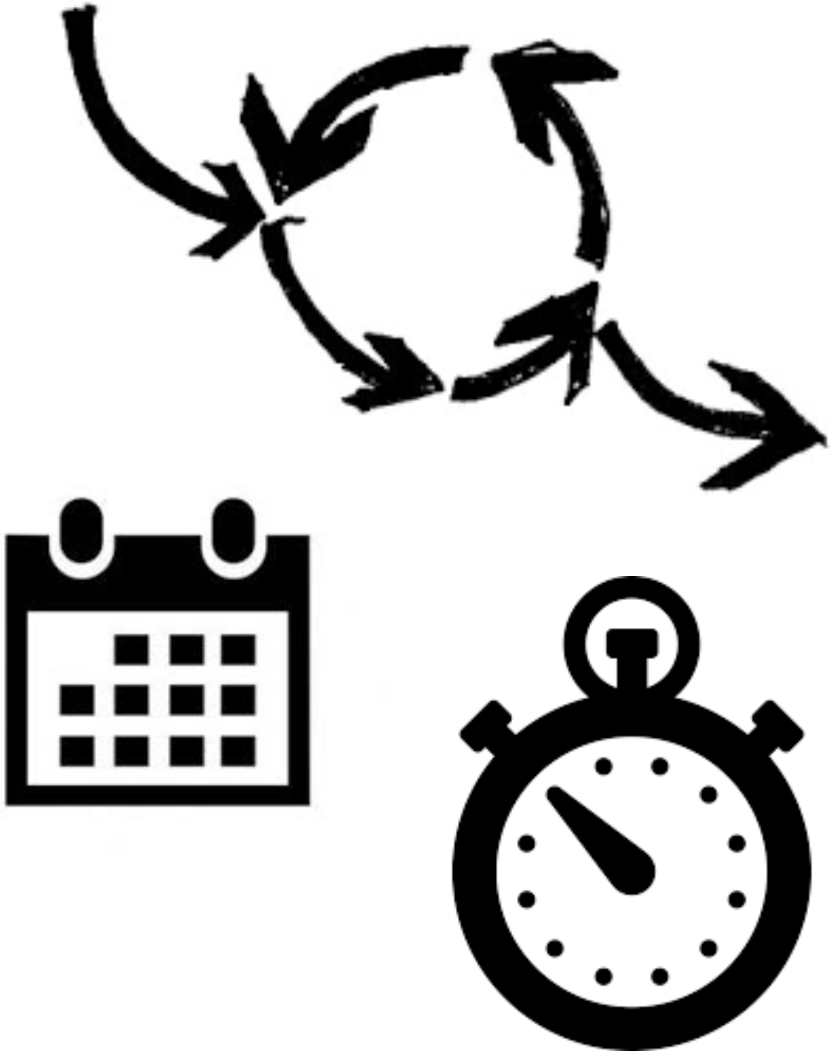
- Number of locations
- Work allocation
  - dependent, distributing
  - independent, isolated, moving
  - expertise-based
- Team structure
  - closely coupled, virtual
  - loosely coupled
  - matrix
- Involvement strategy
  - Modular development
  - Concurrent, parallel development
  - Follow-the-sun
  - Stage-, phase-based handover
  - Division by customization







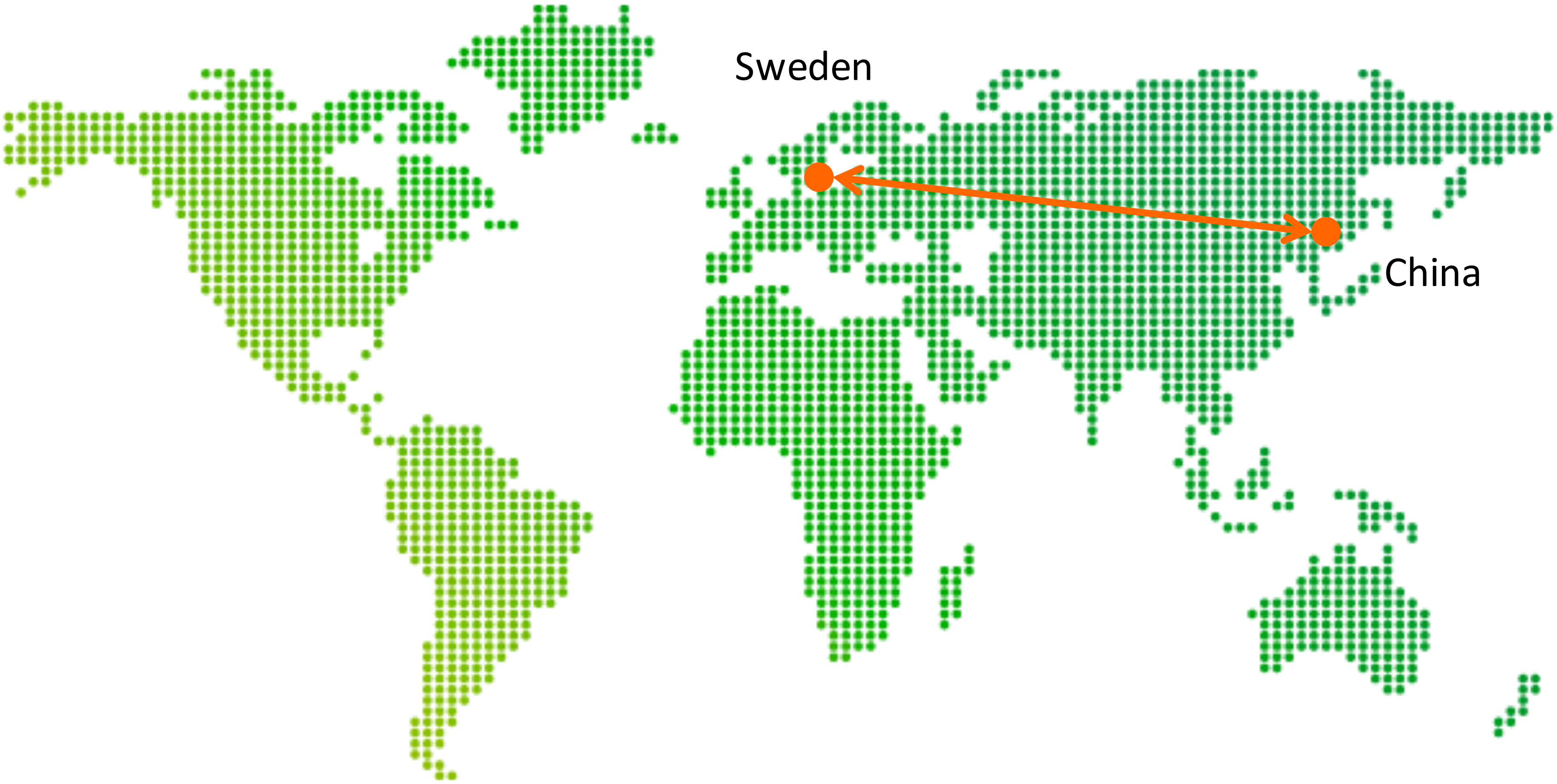
When to source?



When

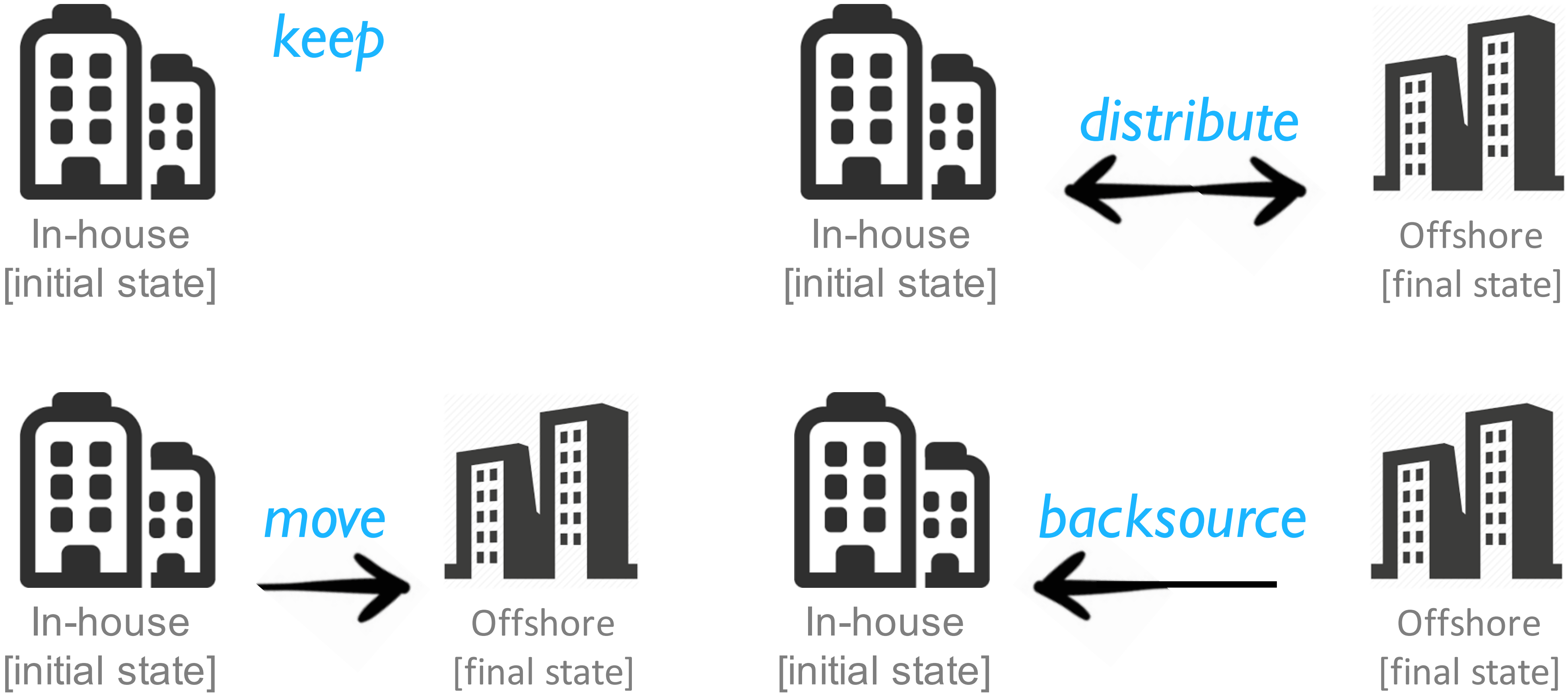
- New development
- Ongoing development (legacy)
- Consequent instances, releases
- In the middle of an ongoing release
- For maintenance
- With sufficient time to ramp-up the project

Example



What happened and why?		What?	When?	Where?	How?
A in SE: Shortage of staff	To get more resources	A: D&I	2000 - 2002	SE to China	Involvement (distributed work)
A distributed between SE&China: Too high dev. costs in SE	To reduce costs	A: D&I	2003 – 2004	SE to China	Transfer of D&I
A distributed between SE&China: New products require resources in SE	To free up resources	A: SPM and V&ST	2005 – 2008	SE to China	Full transfer
A fully allocated in China: a new product (B) is decided to be integrated with product A	To maintain interfaces between components	A+B	2008 – 2009	SE & China	Distributed Integration-centric development
Integration-centric development of the new compound product: Low quality, increase in coordination across sites	To improve the quality and isolate faults across components	A+B	2009 - 2010	SE & China	Distributed modularized development

Alternatives



In-house
[initial state]

keep

In-house
[initial state]

distribute

Offshore
[final state]

In-house
[initial state]

move

Offshore
[final state]

In-house
[initial state]

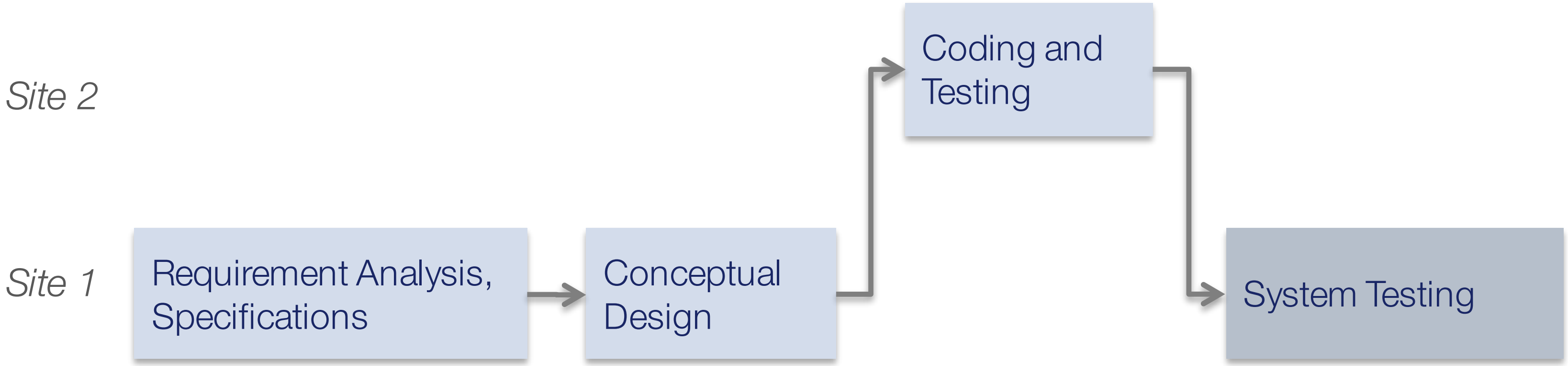
backsource

Offshore
[final state]

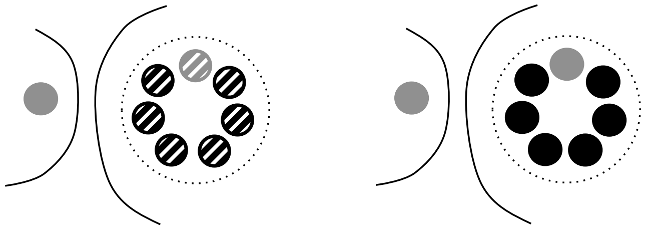
Various ways to do things jointly

Most popular model

By phase

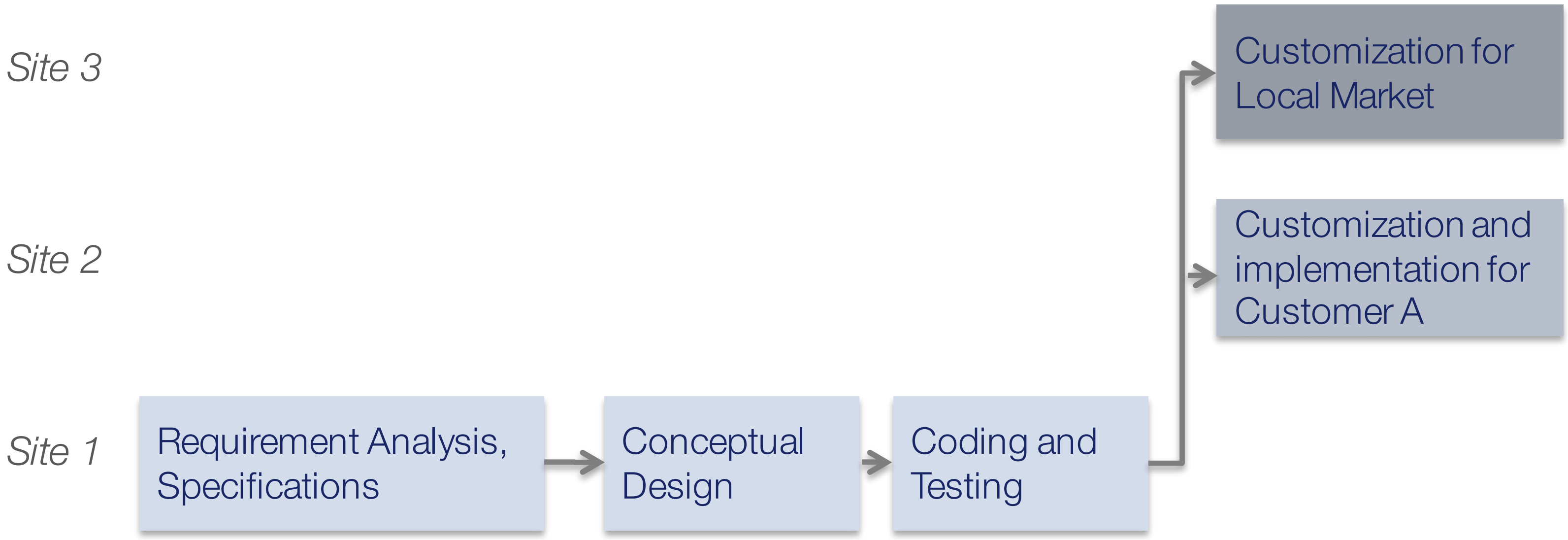


One of the most frequently met collaboration type is sourcing of development tasks (coding) without joint performance



Customization model

By task

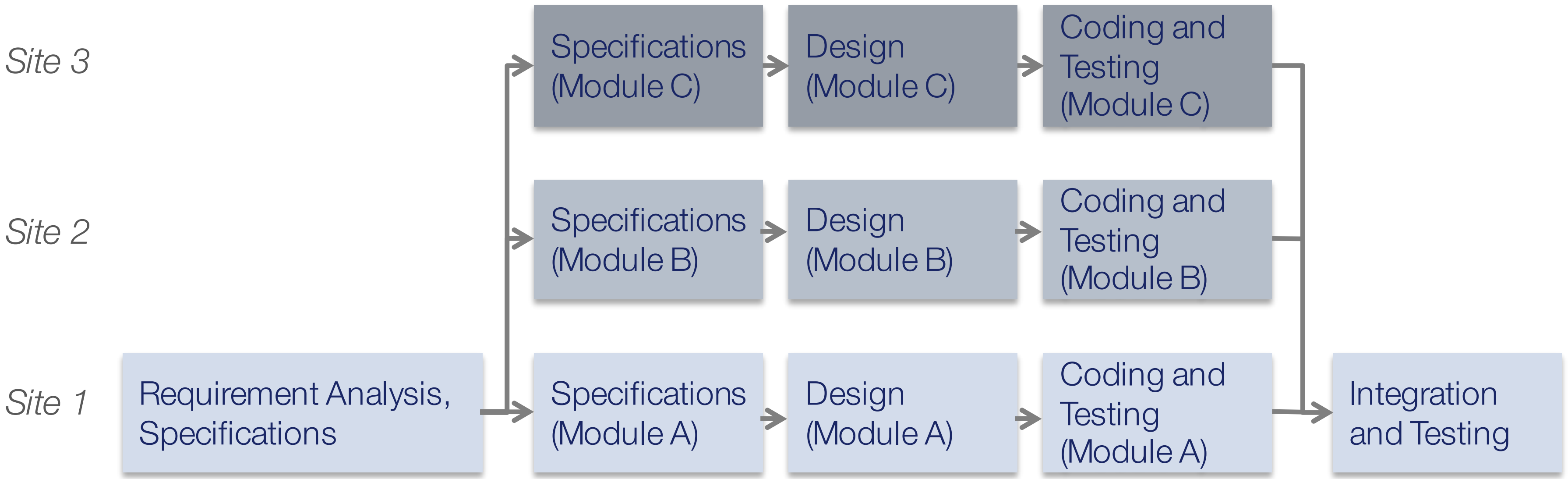

Site 3
Site 2
Site 1
Requirement Analysis, Specifications
Conceptual Design
Coding and Testing
Customization for Local Market
Customization and implementation for Customer A

Globally distributed software development based on customization

REF: "Managing globally distributed teams", In The Handbook of Global Outsourcing and Offshoring by I.Oshri, J.Kotlarsky, L.P.Willcocks

Modularization model

By component


Site 3
Site 2
Site 1
Specifications (Module C)
Design (Module C)
Coding and Testing (Module C)
Specifications (Module B)
Design (Module B)
Coding and Testing (Module B)
Requirement Analysis, Specifications
Specifications (Module A)
Design (Module A)
Coding and Testing (Module A)
Integration and Testing

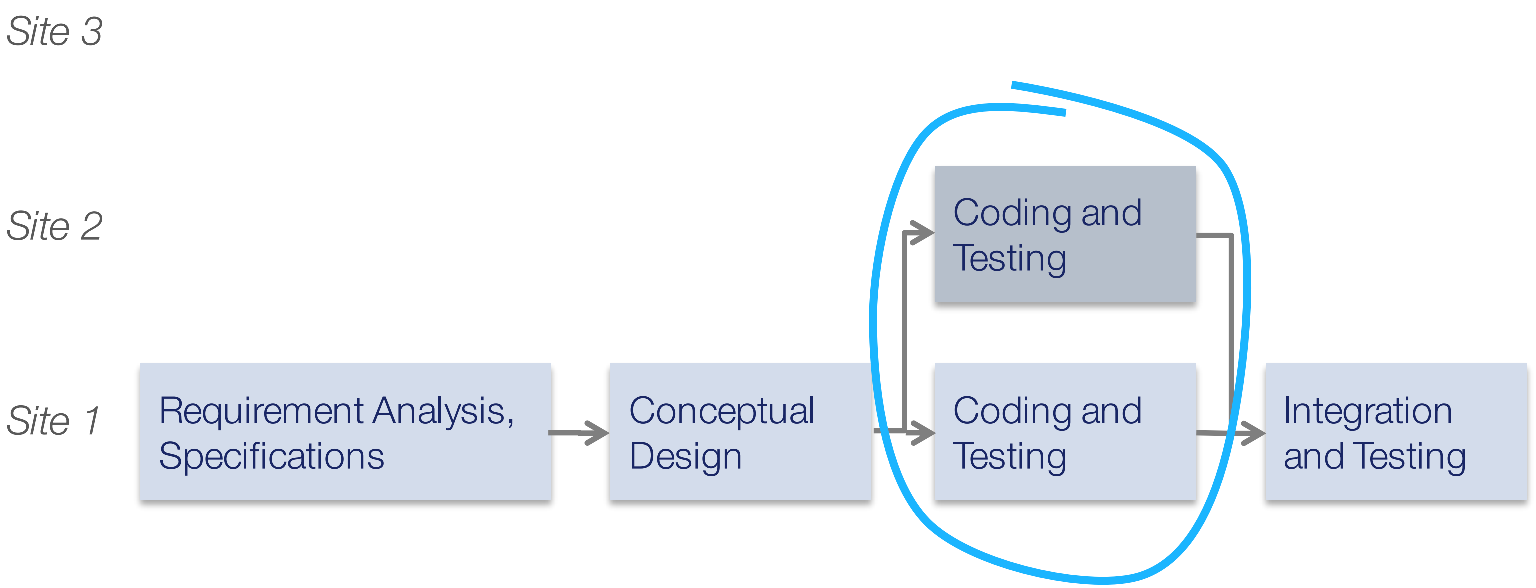
Inter-site coordination for globally distributed software teams:

Strategies aim at reducing the need for inter-site coordination and communications

REF: "Managing globally distributed teams", In The Handbook of Global Outsourcing and Offshoring by I.Oshri, J.Kotlarsky, L.P.Willcocks

Distributed model

Distributed work



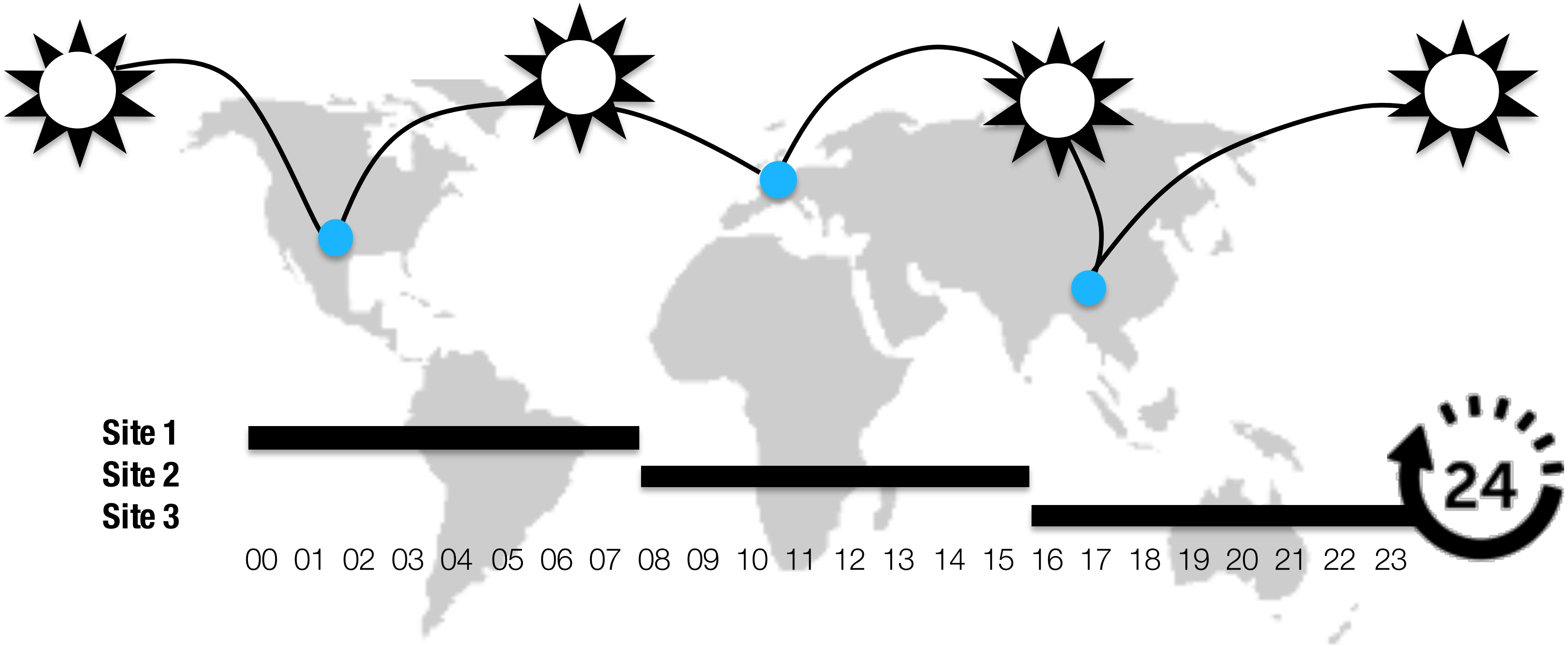
Only well-defined tasks distributed across locations

Tightly coupled work items that require frequent coordination and synchronization should be performed within one site (Lucent Technologies)

REF: "Managing globally distributed teams", In The Handbook of Global Outsourcing and Offshoring by I.Oshri, J.Kotlarsky, L.P.Willcocks
REF: Mockus A. and D.M. Weiss, "Globalization by Chunking: A Quantitative Approach", In: IEEE Software 18(2), 2001, pp. 30-37

Follow-the-sun model

Sequential work



Generally, the 'follow-the-sun' concept is not good for development. However, it works for defect resolution and support

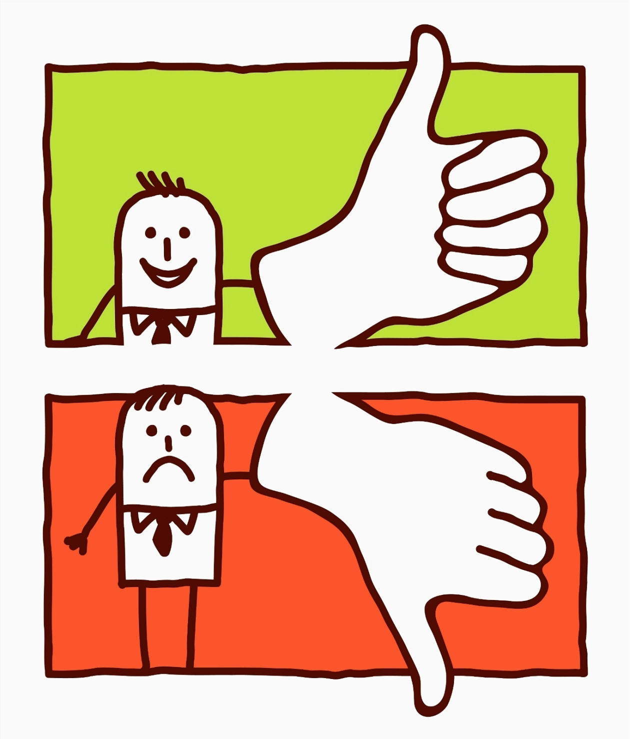
Takes time to build up

REF: H. Holmstrom, E. O Conchuir, P. J. Agerfalk, B. Fitzgerald „Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance ", In Proceedings of the ICGSE conference Oct. 2006, pp.3-11

So which GSE projects fail and where are the benefits?

Company	Projects	Type of work	Sourcing	Sites	Result	Savings/losses
HP	1 collaboration	Remote support	Insourcing	US, Ireland, India	+	10× savings compared to sending an engineer to the customer's site
	1 collaboration	Remote support	Insourcing		+	Further 3× savings
Nokia	3 projects	Basic-complexity projects	Outsourcing	Finland, US, Central and Eastern EU, India, China, others	+	N/A
	6 projects	Moderate- complexity projects	Outsourcing		+	
	4 projects	Complex projects	Outsourcing		+	
	3 projects	Moderate- complexity projects	Outsourcing		–	
	2 projects	Complex projects	Outsourcing		–	
Various	Many	Telecom and automotive	N/A	N/A	+	10–15% savings after 2–3 years
Schlumberger	2 projects	Complex and knowledge-intensive development tasks	Outsourcing	N/A	–	N/A
N/A	1 project	Development and maintenance of financial software	Outsourcing	2 sites in US, Ireland	+	N/A
Nokia	2 projects	Test automation	Outsourcing	Finland, India	+	N/A
	1 project	Test automation	Outsourcing	Germany, China	+	N/A
N/A	1 project	Improvement of a large legacy banking application	Insourcing	Finland, Eastern European country	–	N/A
N/A	1 collaboration	Complex system development	Outsourcing	Norway, India	–	N/A
Phillips	>200 projects	Consumer- electronics product software development	Various	Asia, Europe, India	–	2× to 3× costlier than collocated development
N/A	1 project	Continuation of payroll system development	Outsourcing	Norway, Russia	+	35–40% savings
IBM	2 projects	Web application development with follow the sun approach	Insourcing		–	Significant decrease in gross profit
	1 project		Insourcing		–	Significant net loss with + profit

Conclusions



- Simple, basic projects
- Project with well-defined processes and deliverables that require little management
- Fewer sites, larger benefits

- Complex projects that require great expertise
  - Developing embedded software
  - Evolving and maintaining legacy systems
- Domain specific projects
- Highly technical projects
- Follow-the-sun projects