Planning for Reuse

(based on some ideas currently being discussed in LHCb)

John Harvey
CERN EP-ALC

- Obstacles to reuse
- Process for reuse
- Project organisation for reuse
- Project phases
- Plans for immediate future
- Points for discussion

Obstacles to Reuse

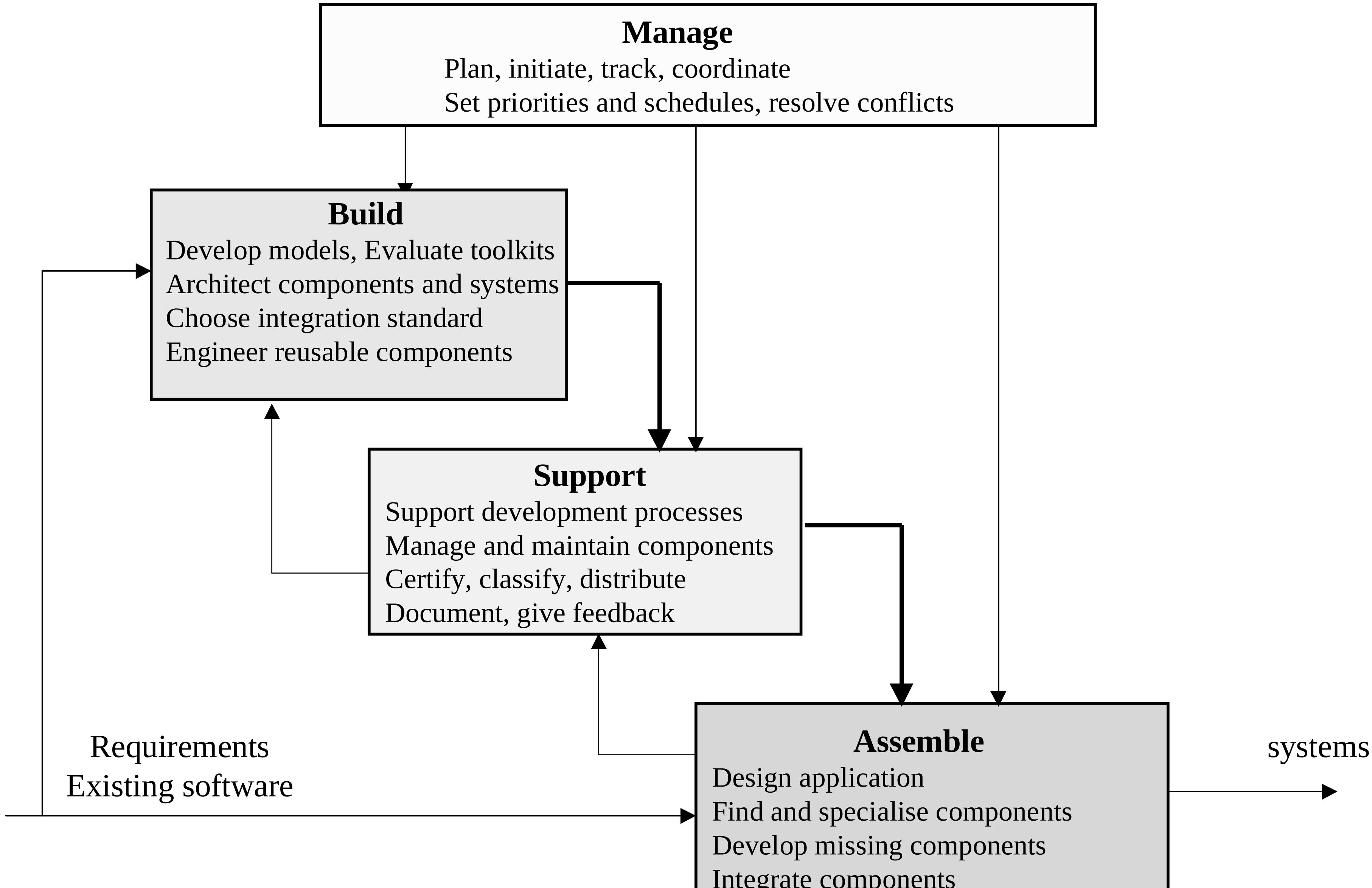
- Engineering
  - poor design and implementation
  - lack of flexibility in components
  - too broad functionality in components
  - poor access to component libraries
- Process
  - role of architect not defined
  - technical review procedures fail to address reuse
  - from existing software, identify what could be packaged for reuse

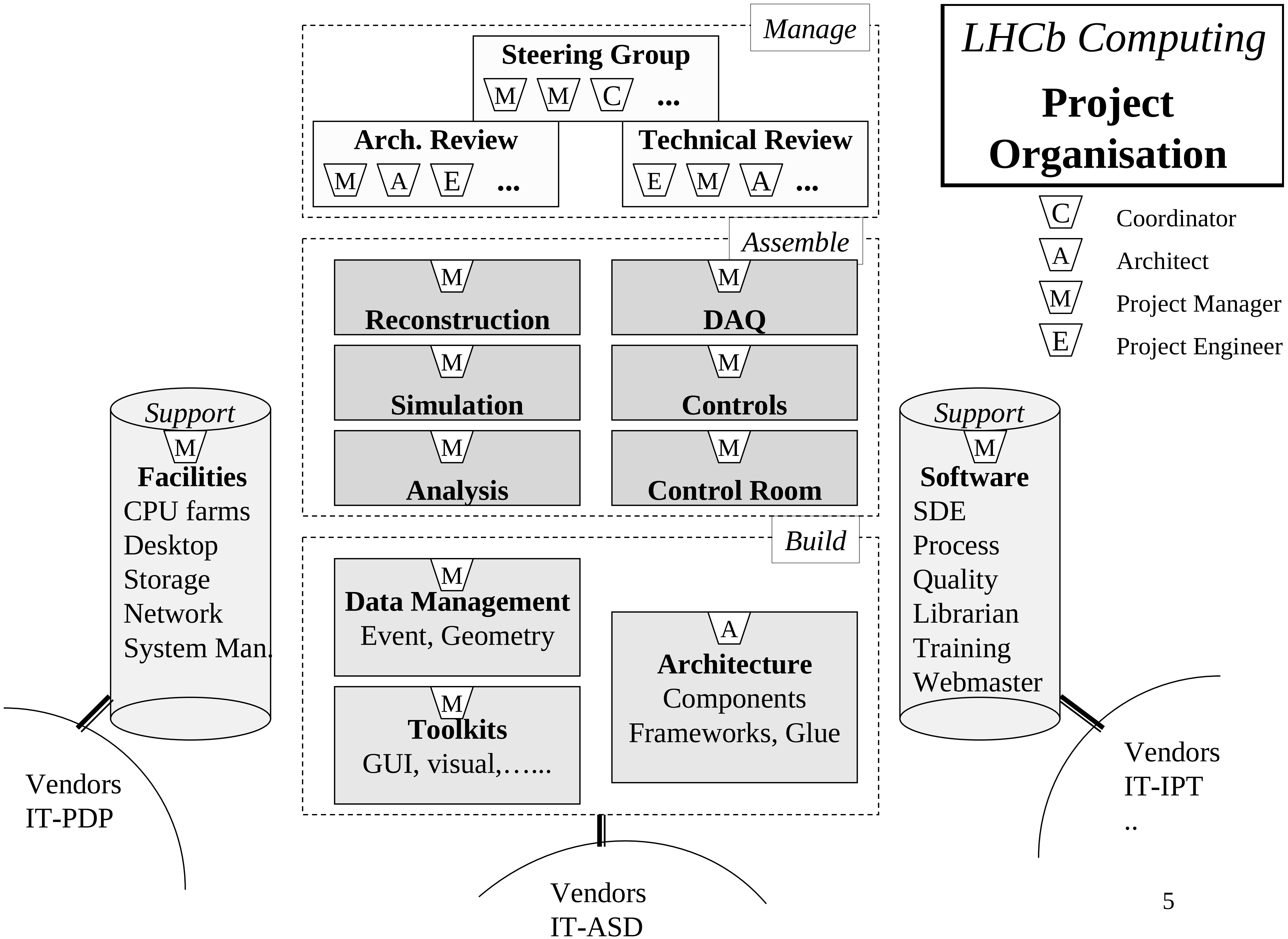
Obstacles to Reuse

- Organisational
  - reuse requires a broad overview
    - we tend to split into separate domains…divide and conquer
    - each project tends to be managed independently of the others
  - culture
    - don’t trust others to deliver what we need
    - fear of dependency on others
    - fail to share information with others
    - developers fear loss of creativity

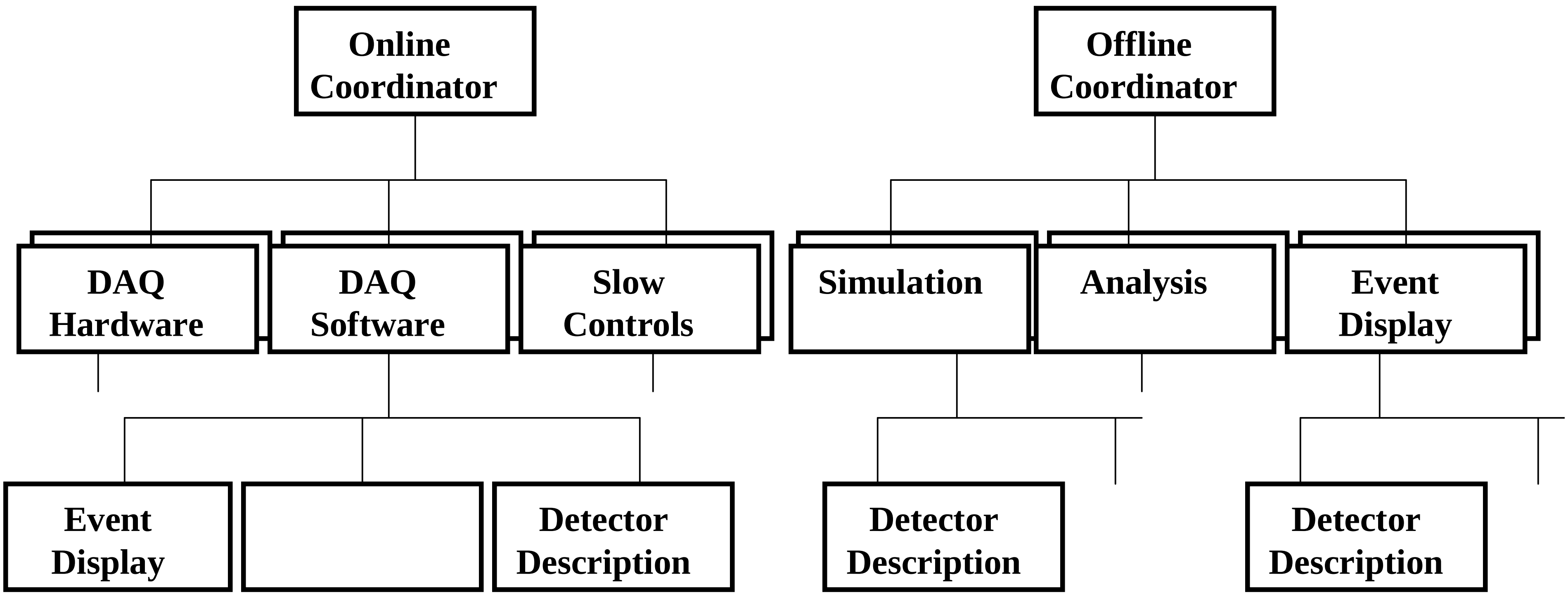
Reuse is a management activity - need to provide the *process* and *organisation* to make it happen

Process for Reuse




LHCb Computing
Project Organisation
Manage
Steering Group
M M C ...
Arch. Review
M A E ...
Technical Review
E M A ...
C Coordinator
A Architect
M Project Manager
E Project Engineer
Assemble
M Reconstruction
M DAQ
M Simulation
M Controls
M Analysis
M Control Room
Support
M Facilities
CPU farms
Desktop
Storage
Network
System Man.
Support
M Software
SDE
Process
Quality
Librarian
Training
Webmaster
Build
M Data Management
Event, Geometry
M Toolkits
GUI, visual,…...
A Architecture
Components
Frameworks, Glue
Vendors
IT-PDP
Vendors
IT-ASD
Vendors
IT-IPT
..

Traditional Project Organisation



Project Phases

‘98

Preparatory Phase

activities : OO migration, prototyping, technology evaluation, requirements analysis, maintain production software

deliverables : functional specifications, architectural design, software process and development environment, organisational structure, decisions on technologies

‘00

Implementation Phase

select, build and test components

‘02

develop applications

stand-alone tests

‘04

Commissioning Phase

Install and integrate applications to make working systems, test in production environment

Operation Phase

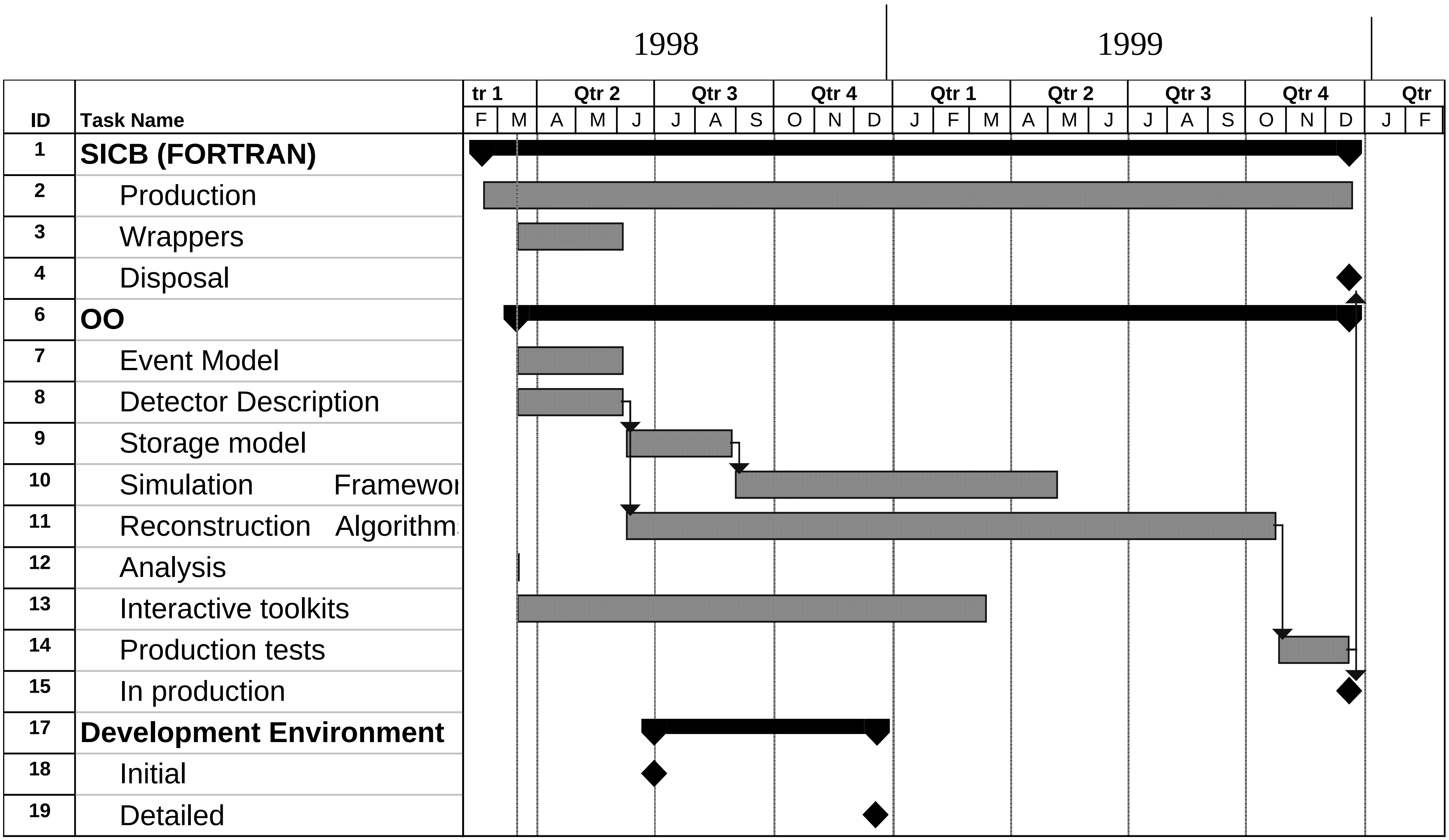
Bring all systems into production 6 months before datataking

‘06

Strategy for next two years

- Dilemma
  - have to keep existing software in production
  - update practices, use new technology for future development
- OO Migration
  - migrate people as soon as possible
  - minimise ‘legacy’ code
  - need to setup a process and environment now
  - train - ‘just-in-time’, CERN training programme
  - algorithm development ongoing
    - *wrap* to be able to use in existing FORTRAN environment
  - evaluate new toolkits
  - start component development (event model, geometry, ….)
- Share ideas, designs, components etc. with other experiments
- Don’t start too many activities, exploit what has been done so far

Schedule for next two years



For discussion?

- Is reuse sufficiently important to put in place a specific *process* and *organisation* that support it?
- Should all aspects of Computing be treated in one project structure?
- Should there be a role for architects and for architectural review?
- What components are out there (other experiments, commercial….) that we can start to use e.g. event model, geometry model,….?

Architecture

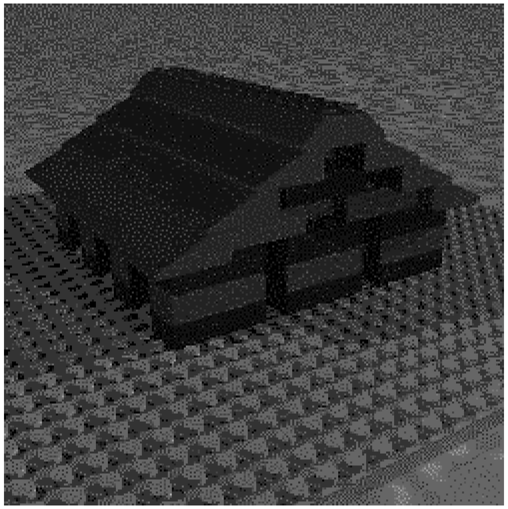
- Definition
  - principles, mechanisms and patterns that define and communicate the structure of a system
- Notation
  - designs communicated using a modeling notation e.g. OOA and OOD
- Benefits of architecture-driven development
  - allows development of understandable systems that are easier to maintain
  - allows applications and ***components*** to evolve gracefully
  - enables developers to achieve a significant level of ***reuse***
  - ***enables components to work well together***

Widely used terms

- Component
  - is a piece of software engineered to be reusable
  - should be cohesive i.e. perform well-defined function
  - have a stable public interface
- Component systems
  - groups of related components providing more complete set of functionality
- Layering
  - components can be built from lower-level components
- Frameworks
  - implements core functionality of domain architecture, provides mechanisms to shape all applications developed using it
- Interface Glue (Integration Technology)
  - user's view of a component system
  - should conform to an appropriate architecture/industry standard
  - should not expose too much to permit future evolution
  - involves negotiation between architect, engineer and user

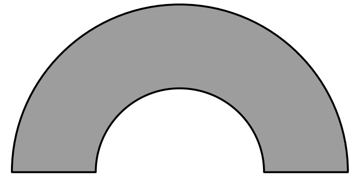
Industrial Figures on Components

- Must be used at least 3-5 times to cover cost of creating and supporting
- costs ~ twice as much to create and support a re-usable component as opposed to implementing it in a given application
- costs ~a quarter as much to reuse a component as it does to develop one from scratch
- takes ~3 years before benefits of reuse become significant

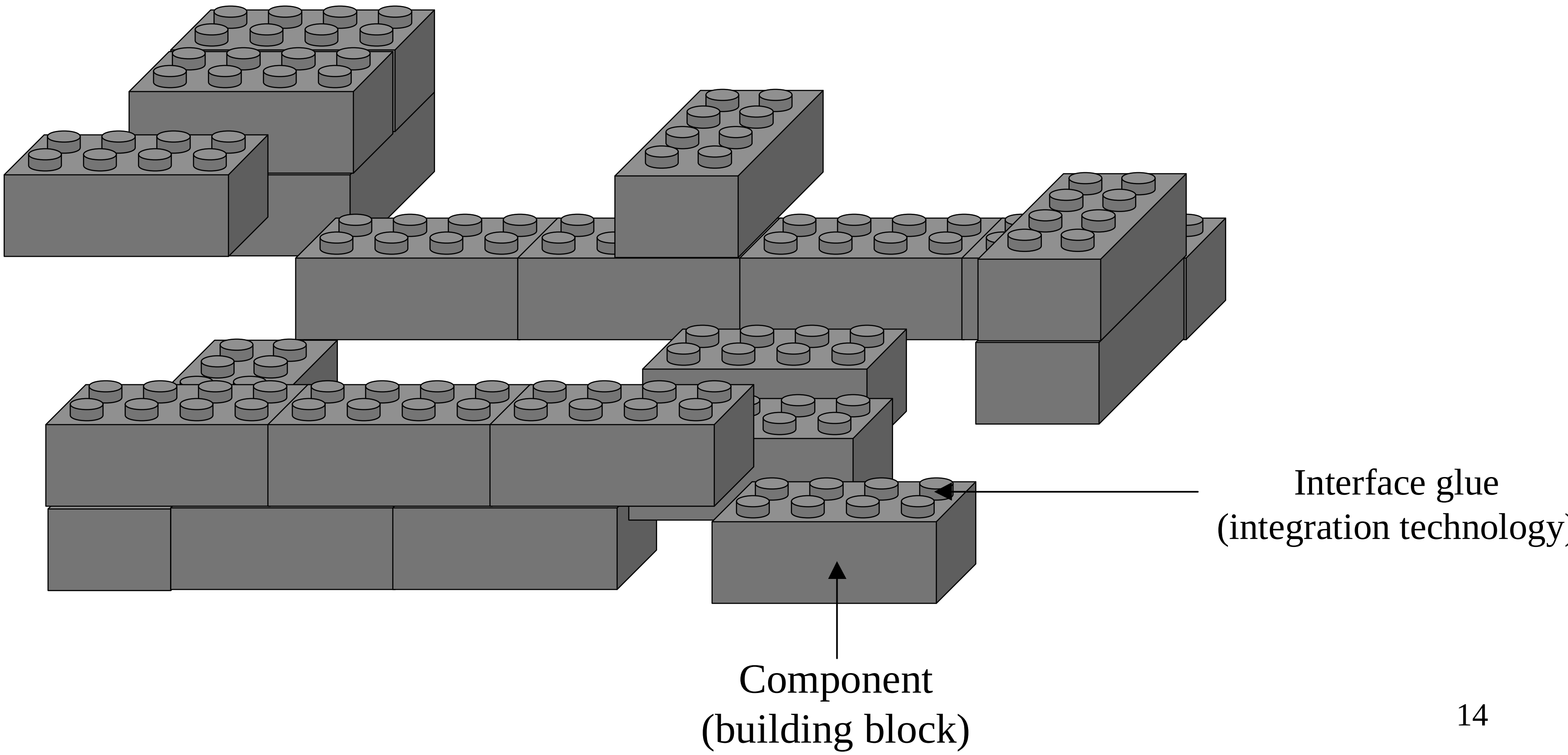


Lego® bricks

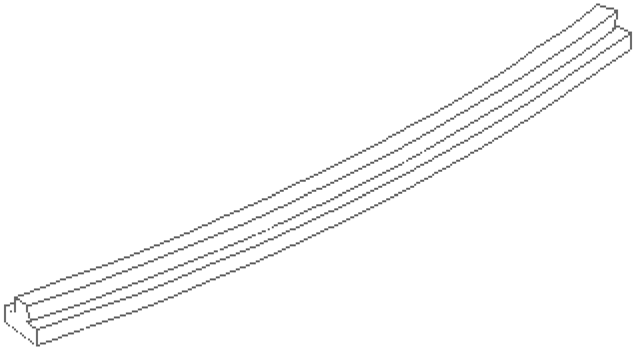
® proprietary



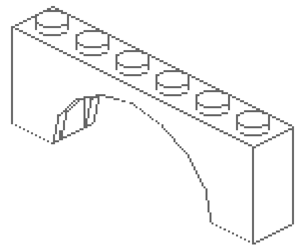
Fischer Technik

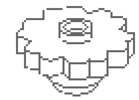


Application Domains

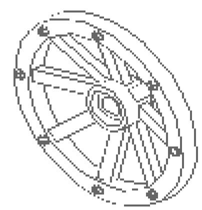


Train Track Curved Inside

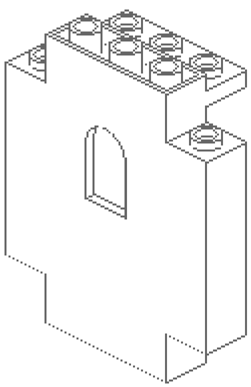




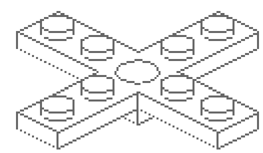
Flower



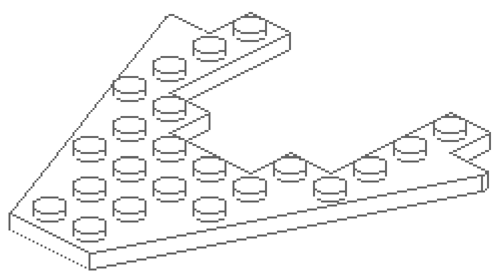
Wagon Wheel



2 x 5 x 6 Castle Wall



4 Blade Helicopter Rotor



8 x 8 Space Wing