



Software Technology 02

Software Development Process and Life Cycle

What is a Software?

The goal is to make a Software, but is it a

- Product?



- Service?



- Infrastructure?



History of Product Handling and Optimization

We force software to be a Product →

- Division of Labor
- Mass Production
- Interchangeable Parts
- Specialization →

The Assembly Line / Production Line

- Predictability

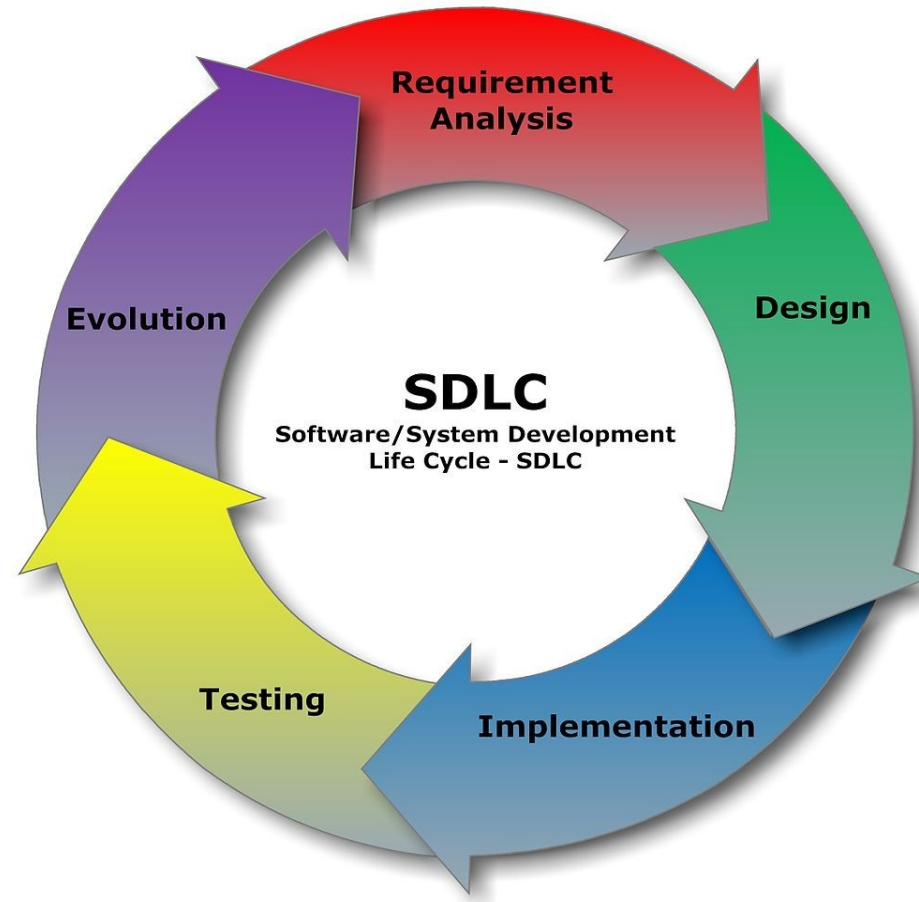


Assembly Line Features

- Consists of Phases
 - Phase uses Results of Previous Phases
- High Quality (involves separate QA phases)
 - QA checks, but quality comes from ...?
- Based on Customer Requirements
- Predictably meets
 - Time Schedules
 - Cost Estimates

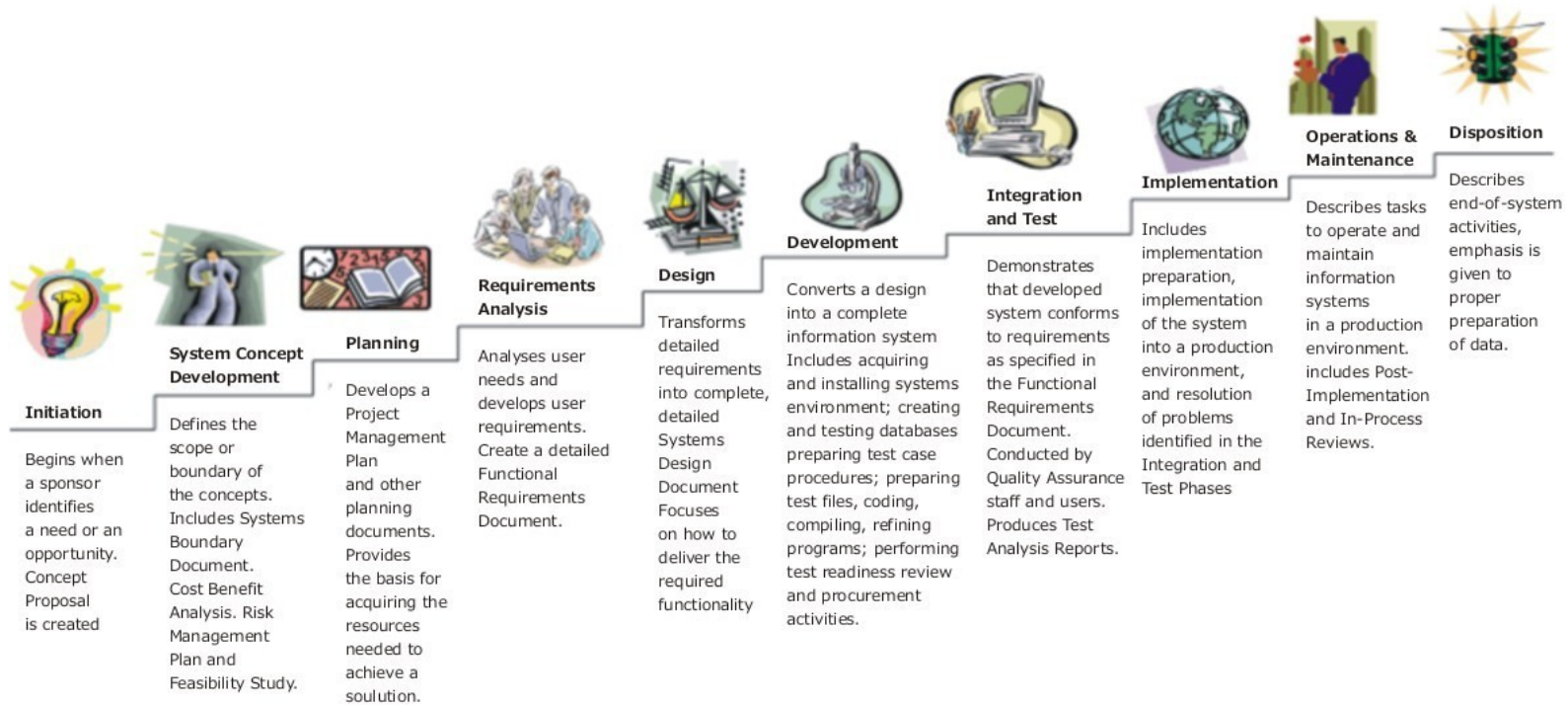


SDLC



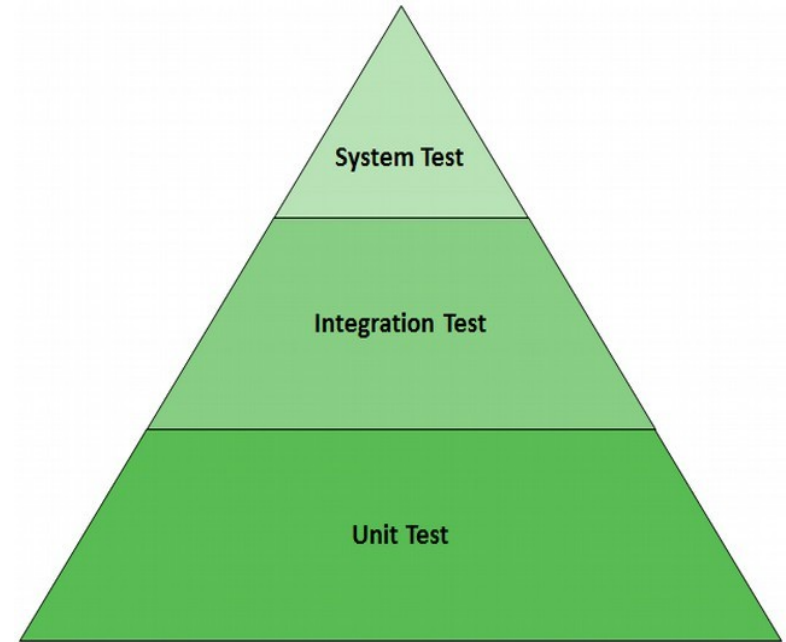
Systems Development Life Cycle (SDLC)

Life-Cycle Phases



Testing (QA)

- Unit / Integration (+interface) / System testing
- White-box / Black-box / (Gray-box) testing
- Smoke testing
- Acceptance testing
- Regression testing
- Alpha / Beta testing
- Usability / Accessibility testing
- Performance testing
- Destructive (Stress and Crash) testing
- Security testing
- A/B testing

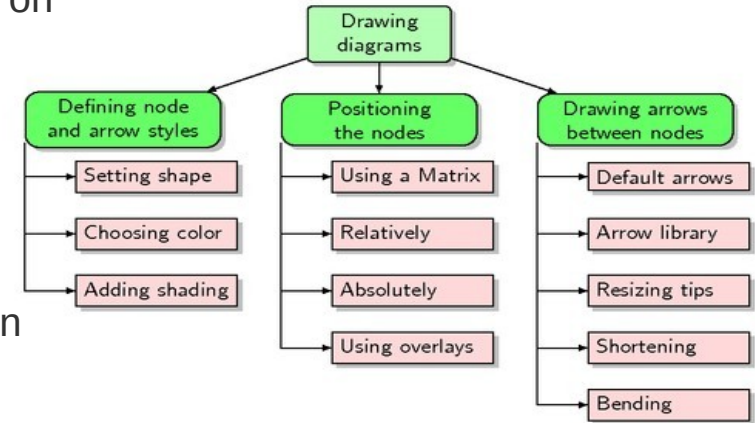


WBS

Measurement and control is important, but can only be done on manageable parts →

- **Work Breakdown Structure**

- Hierarchical (**Tree**) structure
- Contains the whole **Scope** of the Project
- Only tells scope, not schedule or detailed execution plan (resource allocations)
- Breakdown is **Complete** (parent element equals all leaves added)
- Declares acceptance criteria / **outcomes**, not actions
- Breaks down until work unit
 - Can be reliably **estimated**
 - Is **measurable**
 - Fits in a reporting period



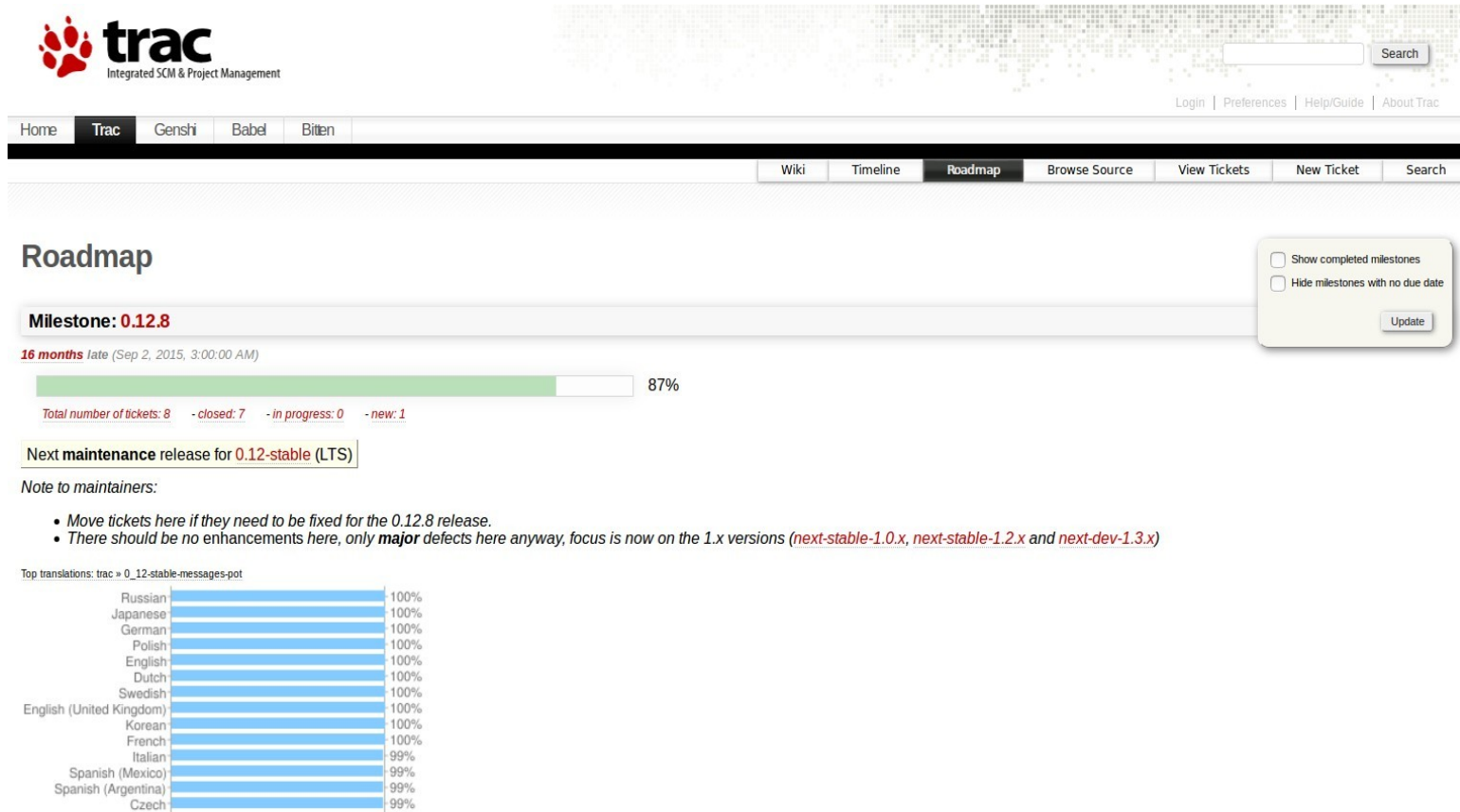
Top-down vs Bottom-up Design

- WBS is a Top-down approach
 - Focuses on high-level features
 - Delivers value (can give the highest value possible to the client / user)
 - Keeps focus on solution requirements
- While Bottom-up can be better in
 - Find simplest / cheapest solutions
 - Reuse existing technology and solutions
 - Find most efficient composition of already made stuff



Probably Top-down with good sense for technological details works,
but needs manager-engineer cooperation

Tools for WBS and beyond (Trac)



The screenshot displays the Trac web interface. At the top left is the Trac logo with the tagline "Integrated SCM & Project Management". A search bar is located at the top right. Below the logo is a navigation bar with links: Home, Trac, Genshi, Babel, and Bitten. A secondary navigation bar contains links: Wiki, Timeline, Roadmap (which is the active tab), Browse Source, View Tickets, New Ticket, and Search. The main content area is titled "Roadmap". It features a section for "Milestone: 0.12.8" with a progress bar indicating 87% completion. Below the progress bar, it states "16 months late (Sep 2, 2015, 3:00:00 AM)" and provides ticket statistics: "Total number of tickets: 8 - closed: 7 - in progress: 0 - new: 1". A box on the right side of the roadmap section contains two checkboxes: "Show completed milestones" and "Hide milestones with no due date", with an "Update" button below them. Below the milestone section, there is a note about the "Next maintenance release for 0.12-stable (LTS)" and a "Note to maintainers:" which includes two bullet points: "Move tickets here if they need to be fixed for the 0.12.8 release." and "There should be no enhancements here, only major defects here anyway, focus is now on the 1.x versions (next-stable-1.0.x, next-stable-1.2.x and next-dev-1.3.x)". At the bottom, a section titled "Top translations: trac » 0_12-stable-messages-pot" shows a list of languages and their translation percentages: Russian (100%), Japanese (100%), German (100%), Polish (100%), English (100%), Dutch (100%), Swedish (100%), English (United Kingdom) (100%), Korean (100%), French (100%), Italian (99%), Spanish (Mexico) (99%), Spanish (Argentina) (99%), and Czech (99%).

trac
Integrated SCM & Project Management

Search

Login | Preferences | Help/Guide | About Trac

Home Trac Genshi Babel Bitten

Wiki Timeline Roadmap Browse Source View Tickets New Ticket Search

Roadmap

Milestone: 0.12.8

16 months late (Sep 2, 2015, 3:00:00 AM)

87%

Total number of tickets: 8 - closed: 7 - in progress: 0 - new: 1

Next maintenance release for 0.12-stable (LTS)

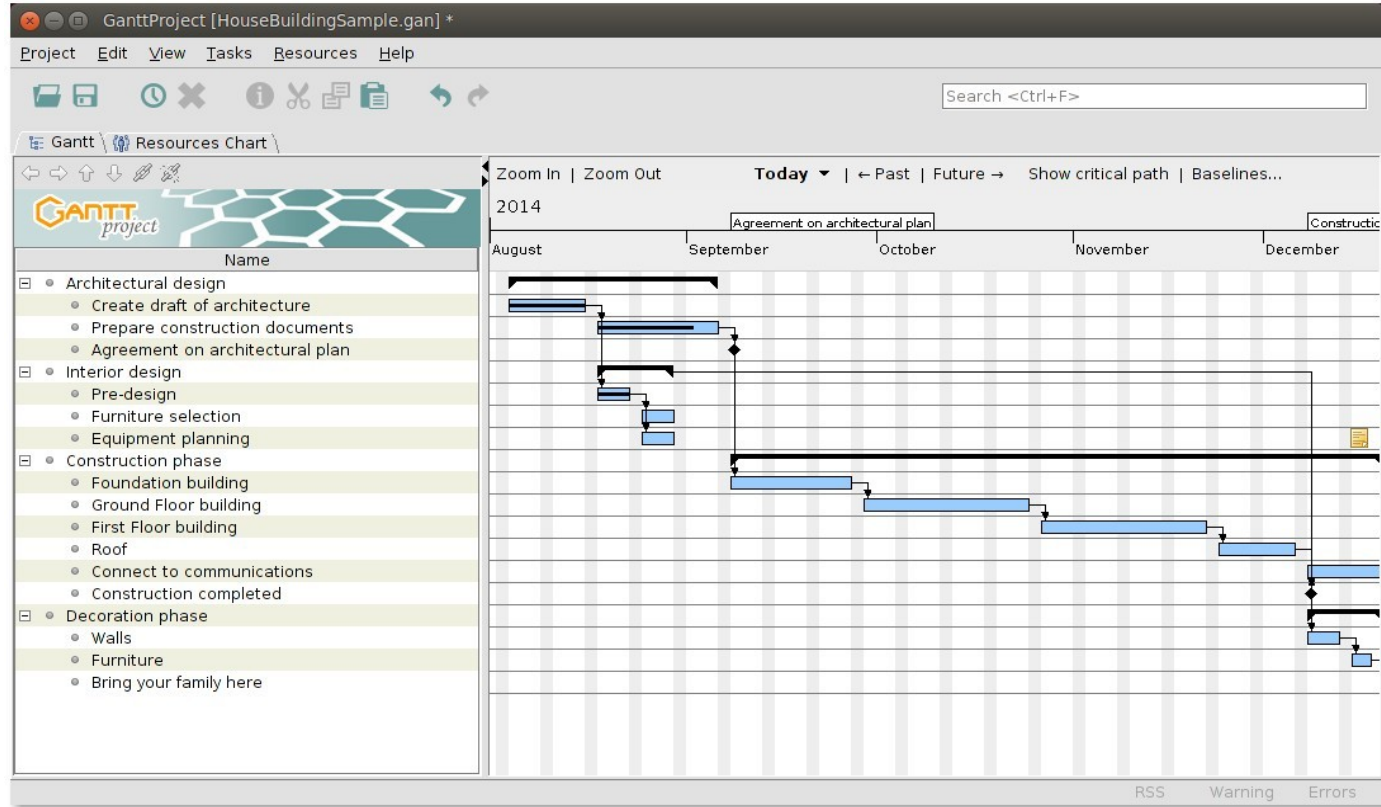
Note to maintainers:

- Move tickets here if they need to be fixed for the 0.12.8 release.
- There should be no enhancements here, only **major** defects here anyway, focus is now on the 1.x versions ([next-stable-1.0.x](#), [next-stable-1.2.x](#) and [next-dev-1.3.x](#))

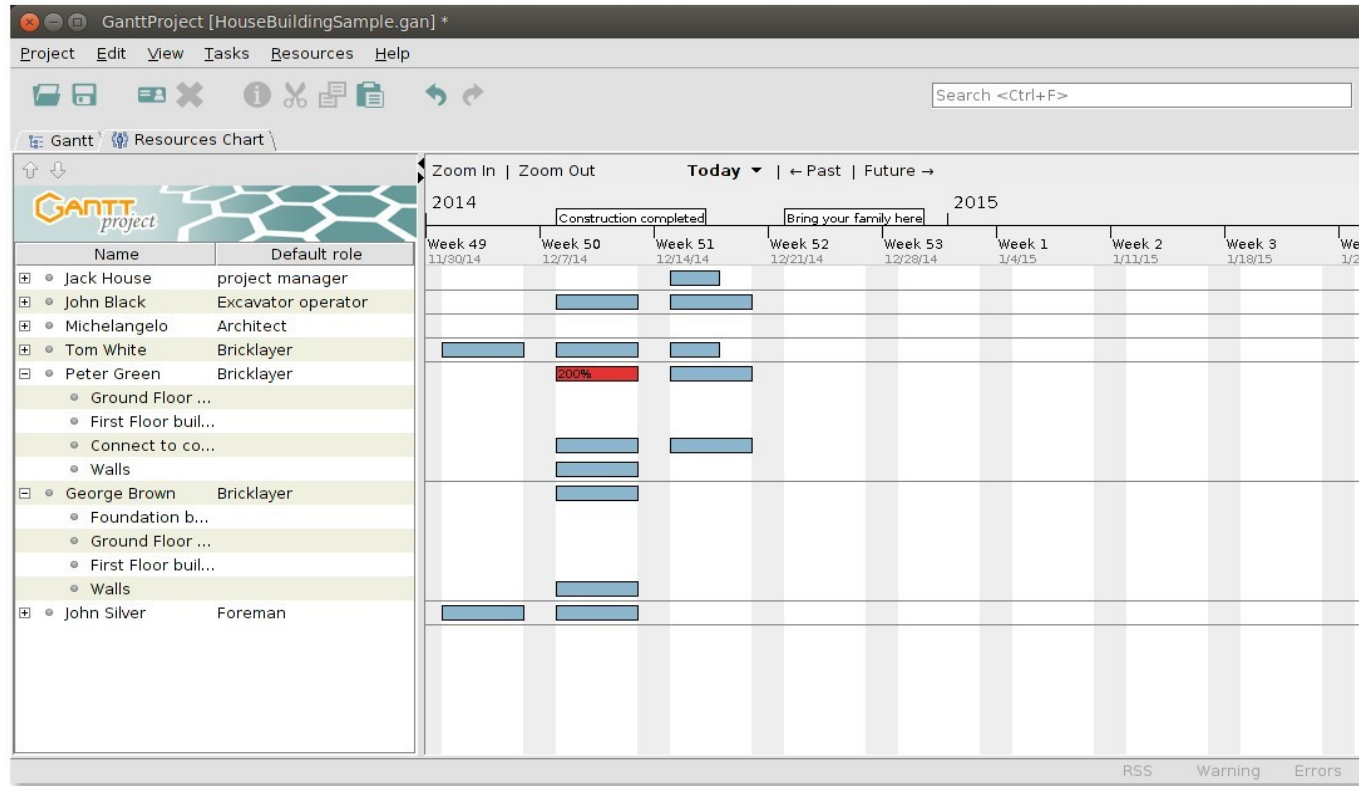
Top translations: trac » 0_12-stable-messages-pot

Russian	100%
Japanese	100%
German	100%
Polish	100%
English	100%
Dutch	100%
Swedish	100%
English (United Kingdom)	100%
Korean	100%
French	100%
Italian	99%
Spanish (Mexico)	99%
Spanish (Argentina)	99%
Czech	99%

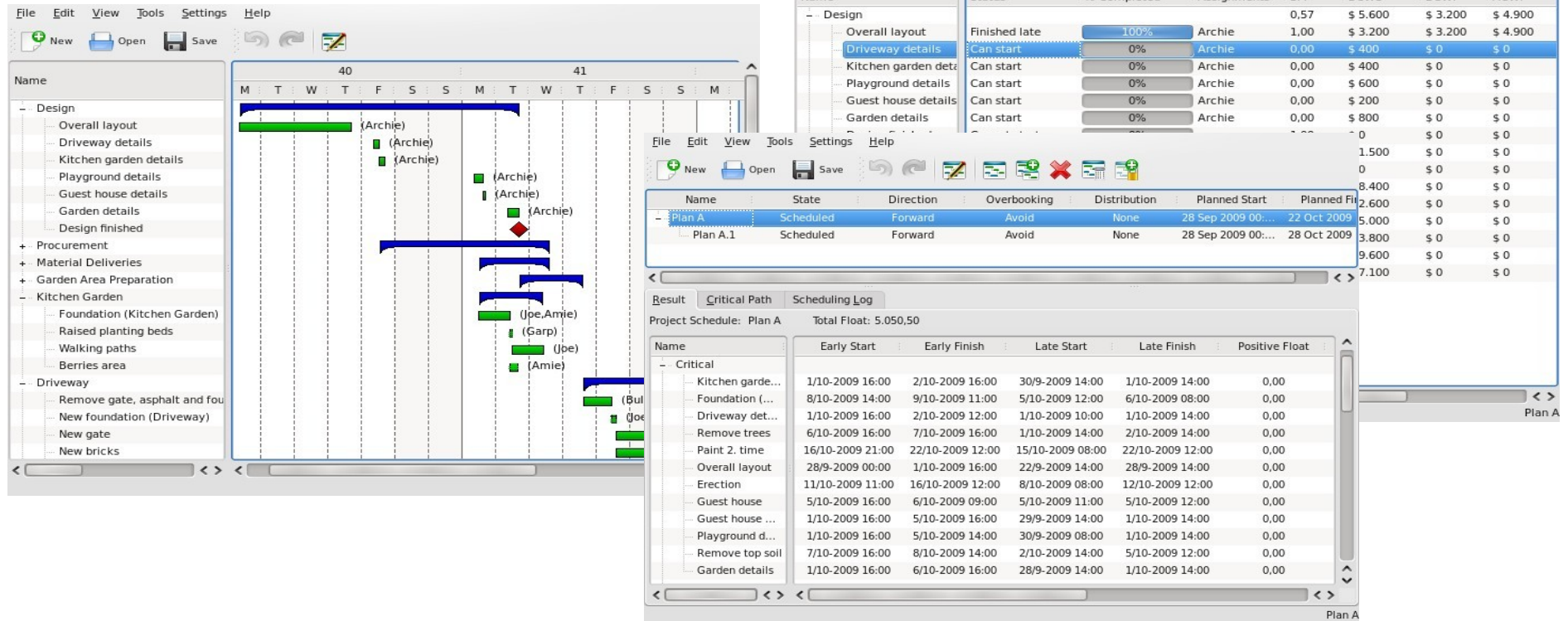
Tools for WBS and beyond (GanttProject)



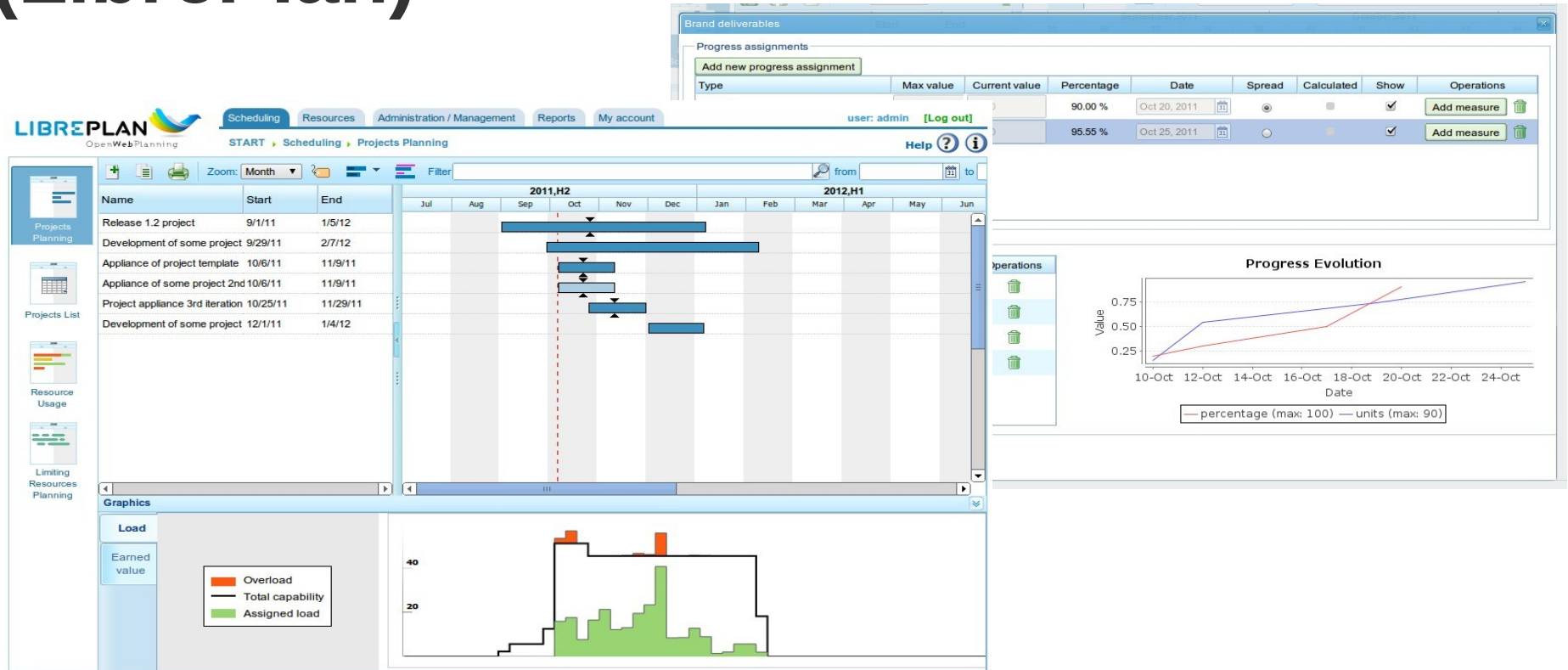
Tools for WBS and beyond (GanttProject)



Tools for WBS and beyond (Calligra Plan)



Tools for WBS and beyond (LibrePlan)



Tools for WBS and beyond (LibrePlan)



Tools for WBS and beyond

For more see:

https://en.wikipedia.org/wiki/Comparison_of_project_management_software

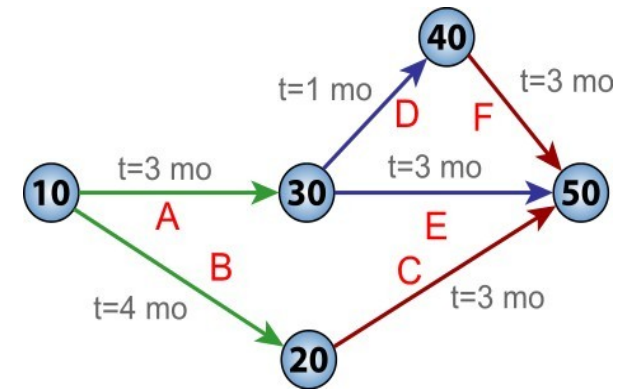
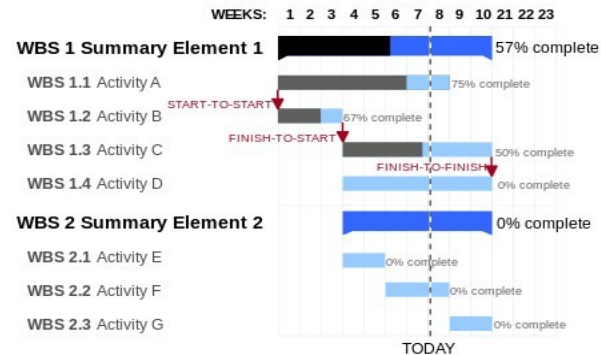
Also see:

Critical Chain Project Management and

Critical Path Method

PERT (Project Evaluation and Review Technique)

Gantt Chart



Other Tools for Project

- Project Execution – *Kanban board*
- Communication – *Chat???* (spoiler alert: NO)
- Version Control – *github, gitlab, gitea*
- Build
- Testing Framework
- Deployment
 - All DevOps

SDLC Criticism

- **Good** in
 - Large projects, complex solutions
 - Detailed, well documented
 - Ease of maintenance
 - Can follow (and certify) standards
- **Bad** in
 - Time and cost is increased
 - Not flexible, cannot handle problems without full-design-first approach
- There are other life cycle approaches
 - Project Life Cycle
 - Application Life Cycle
- Example: SSADM (Structured Systems Analysis and Design Method)
 - Takeaway: Works well on similar problems / projects (in this case information systems)

SDLC vs Assembly Line Analogy

- Assembly Line Features:
 - Every Product is exactly the same we make
 - All work phases are the same, we know
 - How long do they take
 - How much do they cost
 - Workers know everything after training, no further self-development is needed

→ Process risk is very low

NOT TRUE for SW Development!!!

