

**SIEMENS**

---

# Using Software Architecture for Estimation

**Robert L. Nord, Daniel J. Paulish, Dilip Soni**

**Siemens Corporate Research, Inc.  
755 College Road East  
Princeton, NJ 08540 USA**

**© 2000 by Siemens Corporate Research, Inc.**

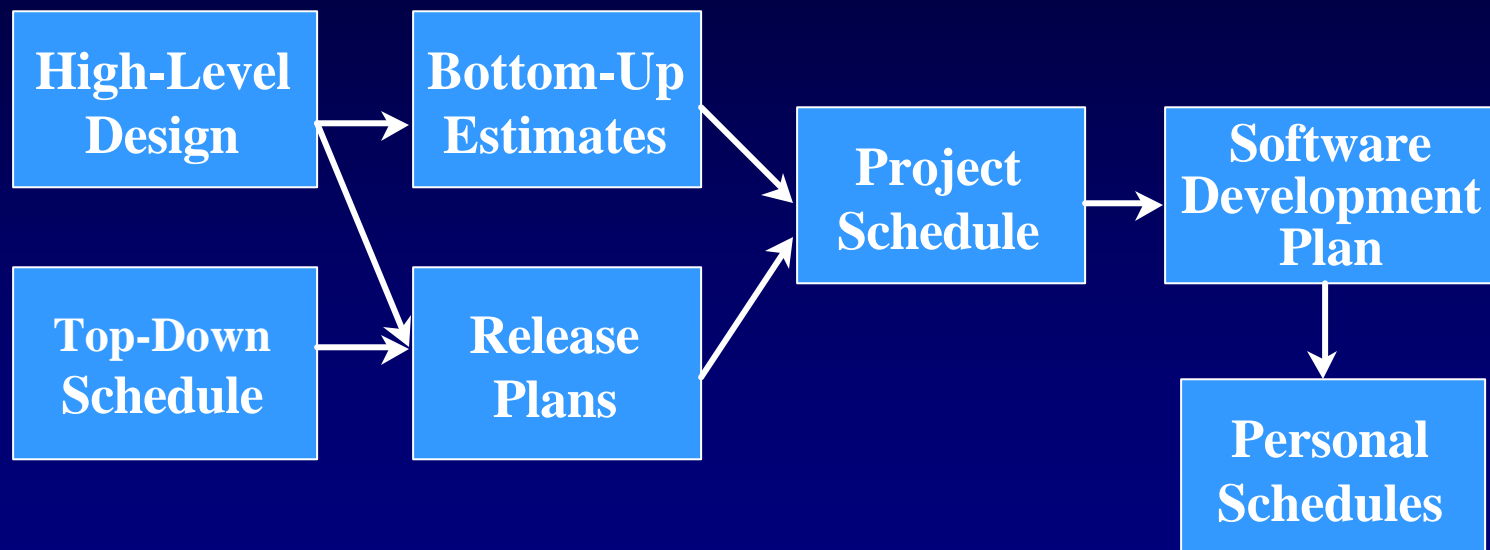
# Realistic Schedules?

**Effort and schedule estimates given in early stages of development can be very inaccurate.**

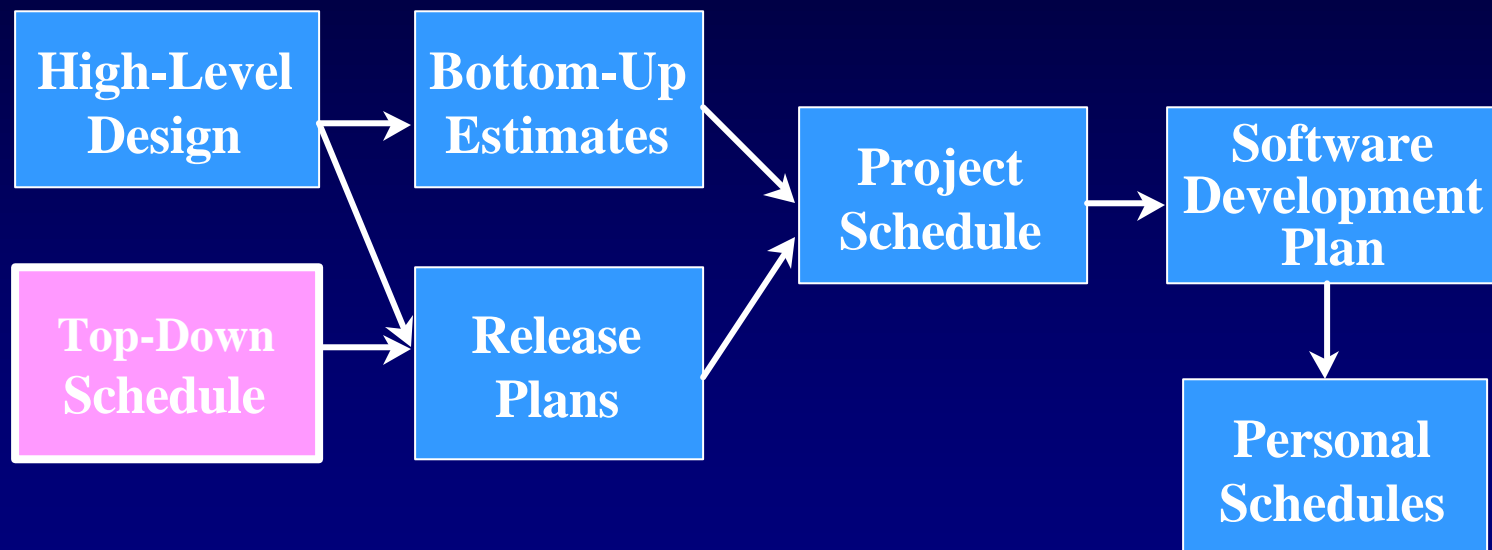
- **Actual effort expended can be 4X the estimates given at the beginning of a project.**
- **Actual effort expended can be 1.5X the estimates given after requirements are complete [Boehm81]**

**Estimates made in the absence of a high-level architecture design have minimal value.**

# Overview of the Process



# Overview of the Process



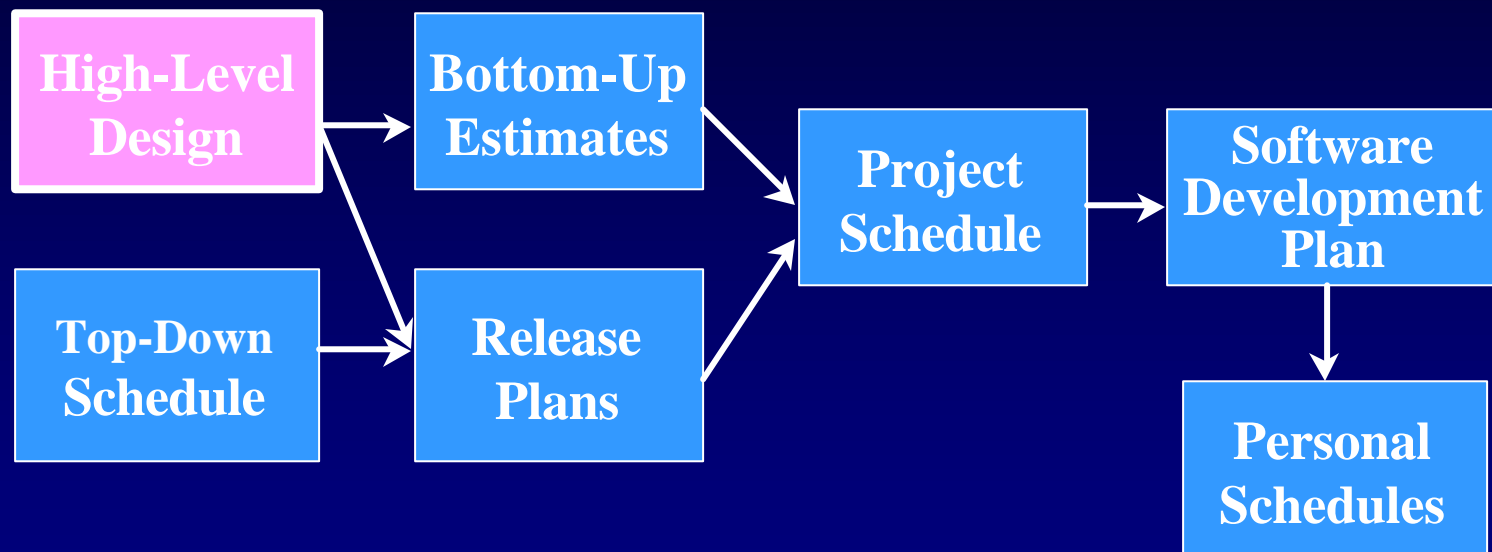
# Top-Down Schedule

**Initiated by the project manager.**

**Preliminary lines of code estimates are inputs to an estimation tool.**

**Outputs provided of effort, schedule duration for major phases of development, and resource profile loading for various types of development skills.**

# Overview of the Process

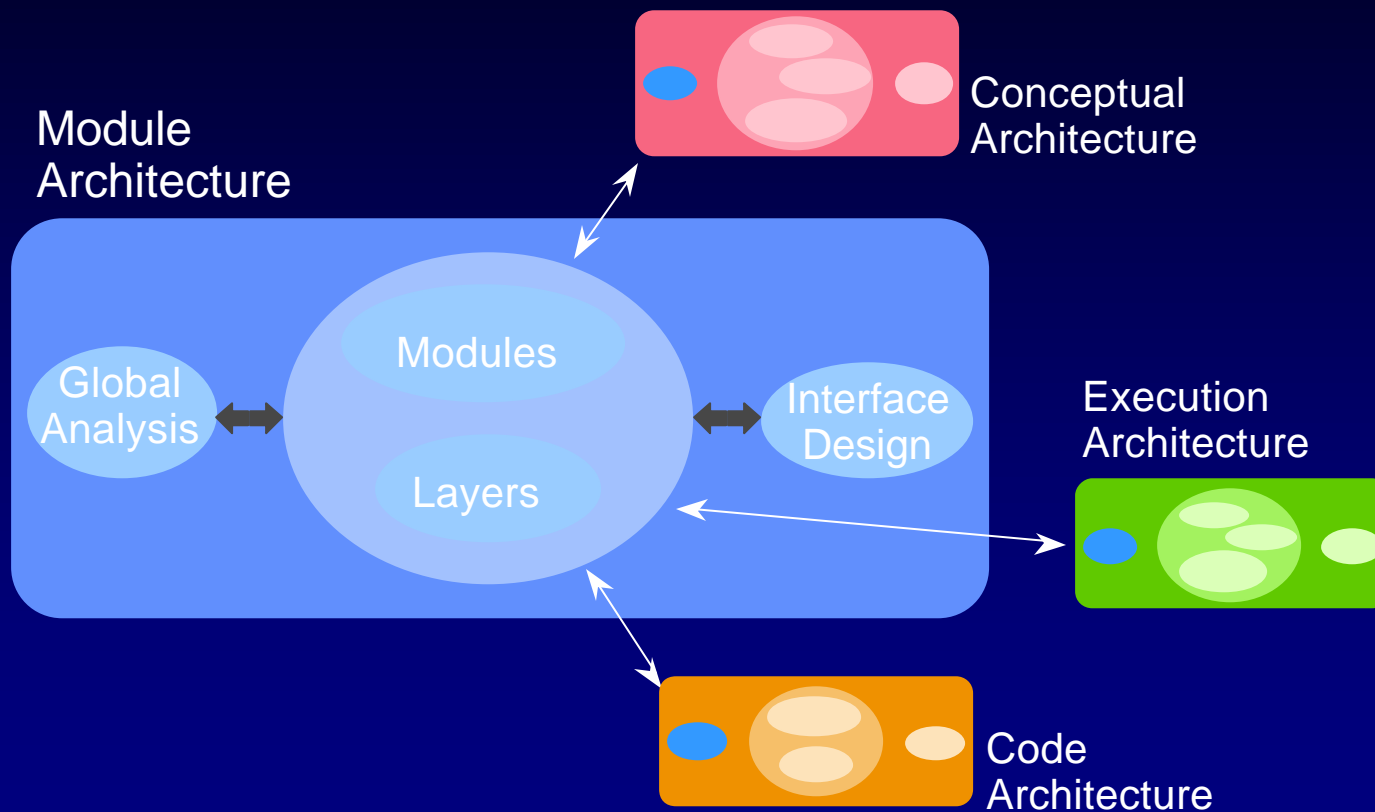


# High-Level Design

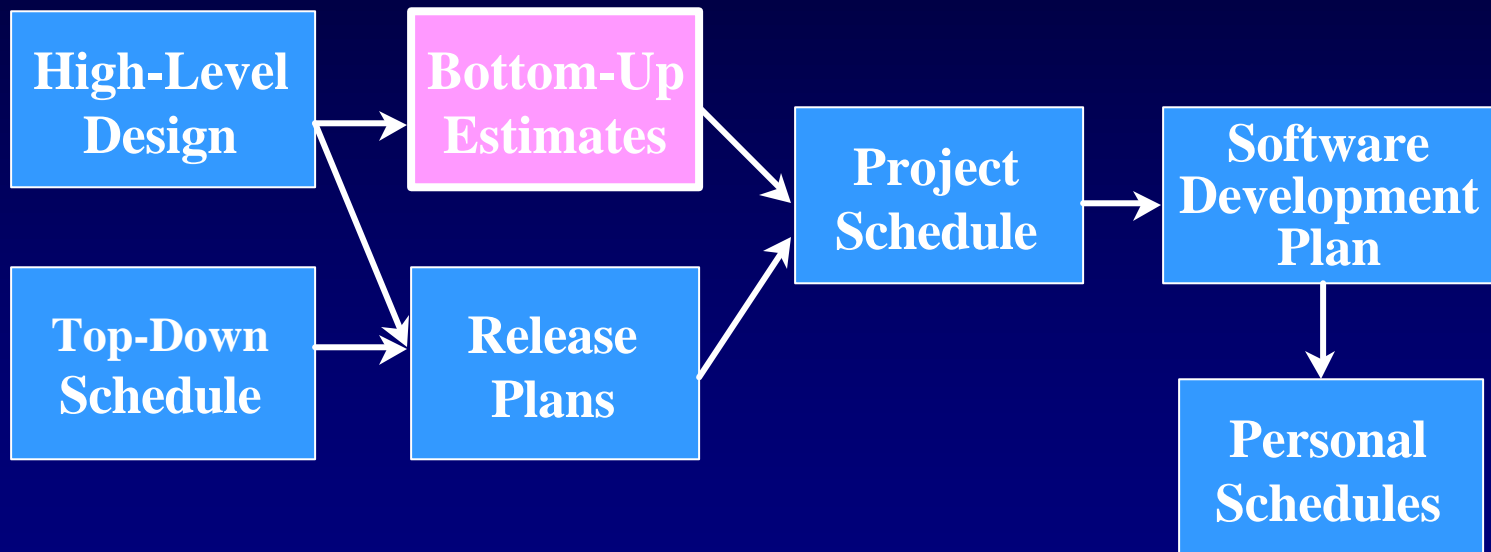
**High-level design of the software architecture is initiated with a small design team.**

**Appropriate level of detail should be developed such that both top management and development team members can get an understandable representation of the architecture.**

# Module Architecture Design



# Overview of the Process



# Bottom-Up Estimates

**Paper design for assigned components, documenting the subcomponents and dependencies.**

**Estimate size, complexity, and effort for each component.**

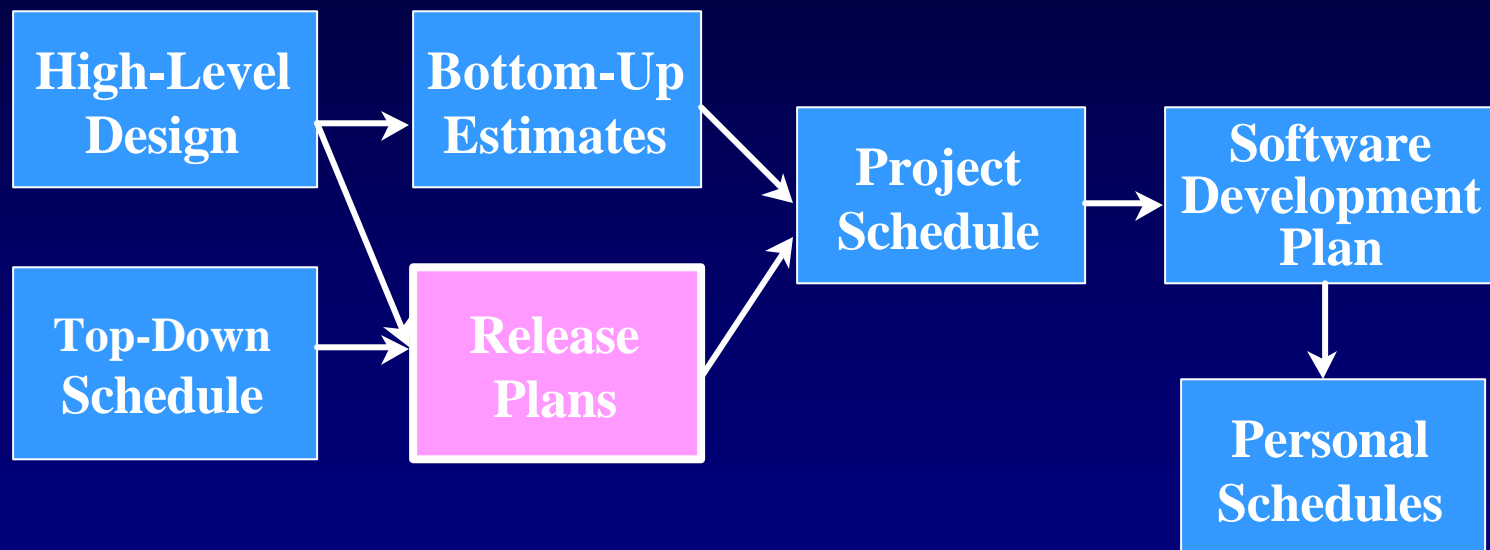
## IS2000 Bottom-Up Cost Estimation

Major work packages

Confidence Complexity Code Size Total Effort Total Effort  
hours staff-yr

<b>Versioned Object</b>	3	5	2000	720	0.3938731
Study Management	3.2	3.4	600	1280	0.7002188
Check In & Check Out	3.7	3.3	1800	1320	0.7221007
Templates	3	5	1000	2400	1.3129103
<b>Schedule Maker</b>	1	5	19500	2920	1.5973742
GUI	2.8	3.6	18300	4960	2.7133479
<b>Communication Syst.</b>	3	4	500	640	0.3501094
Probe Interface	3	3	11500	1580	0.8643326
HW Diagnostics	4	3	400	250	0.1367615
Flat Panel Display	4	3	700	800	0.4376368
All other functions			20000	3500	1.9146608
<b>Totals</b>			76300	20370	11.143326

# Overview of the Process



# Release Plans

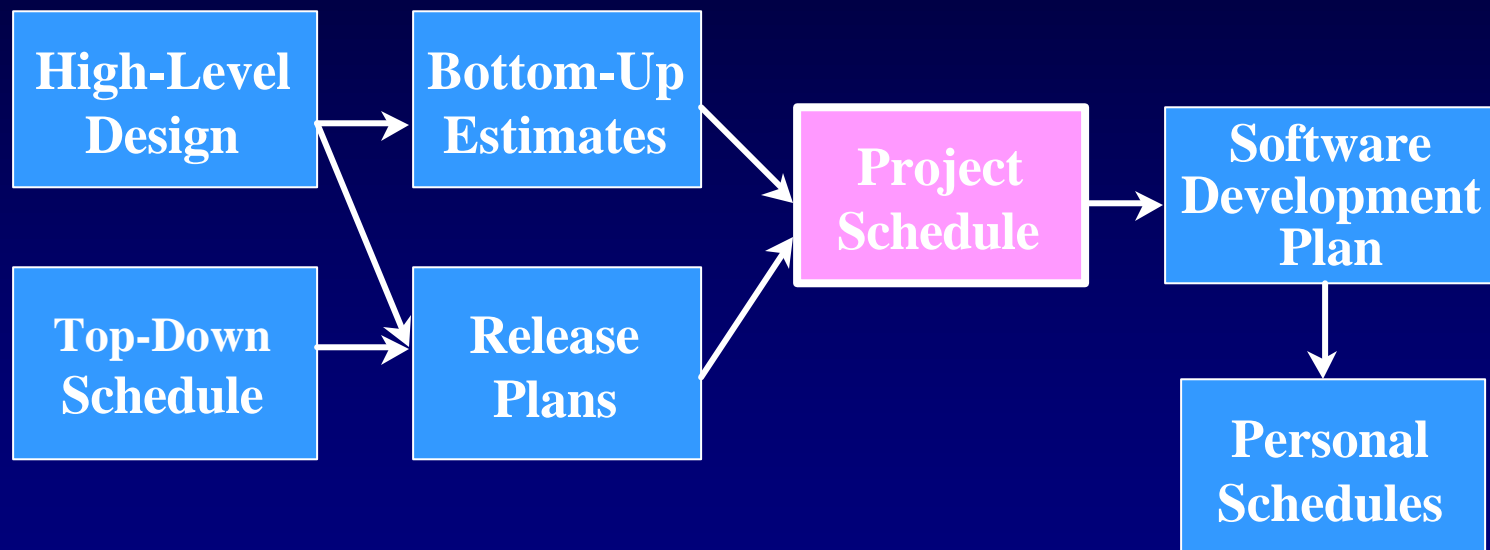
**Incremental releases are planned and implemented as “vertical slices” through the layer diagram.**

**A “build plan” identifies the sequence of implementation for features and components.**

# Example Build Plan

	ER1	ER2	ER3	R1	R2+
<b>Schedule Maker</b>					
Search Consumer Tree for Scheduled Events	ö			ö	
Create a Schedule	ö			ö	
Handle Report Events	ö			ö	
Handle Acquisition Events		ö		ö	
Optimize Acquisitions					ö
Handle set Parameter Scheduled Events		ö		ö	
Display and manual Update of Schedules			ö	ö	

# Overview of the Process

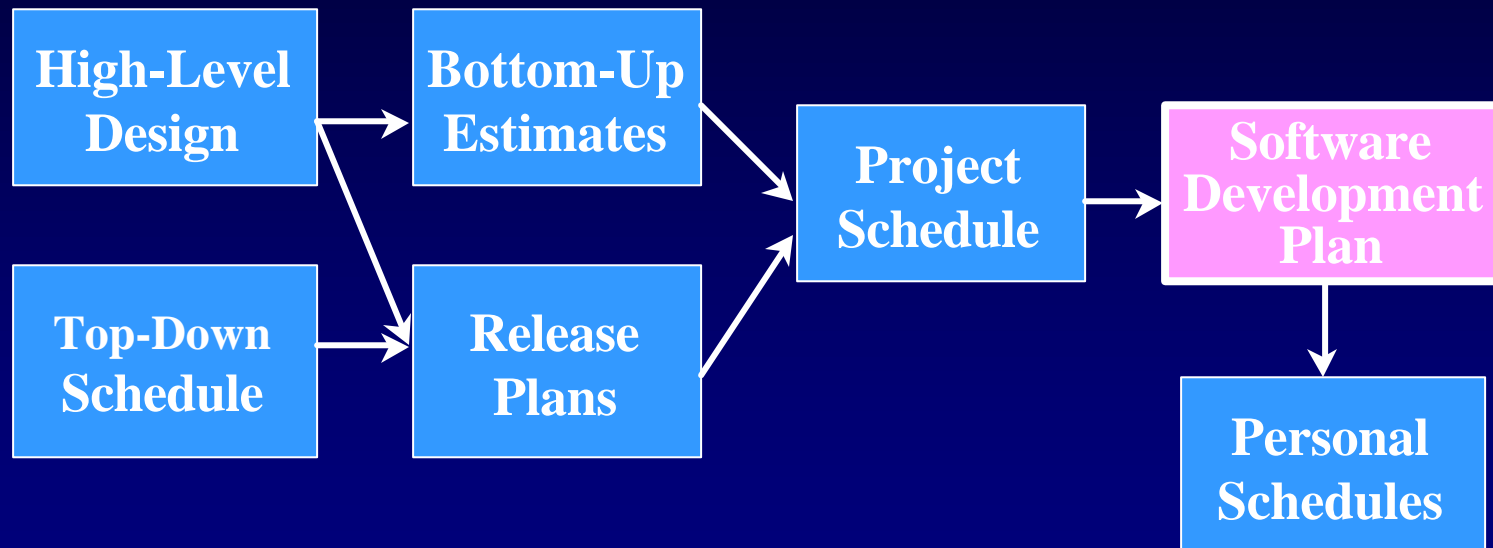


# Project Schedule

**Using the top-down schedule, the bottom-up estimates for each component, the build plan, and knowledge of your development process, you can lay out a project schedule within a skeleton based on subtasks according to the development phases (detailed design, coding, unit testing, bug fixing) for each release.**

**The design and coding for a large component would start early even though its features are not needed until a later increment.**

# Overview of the Process



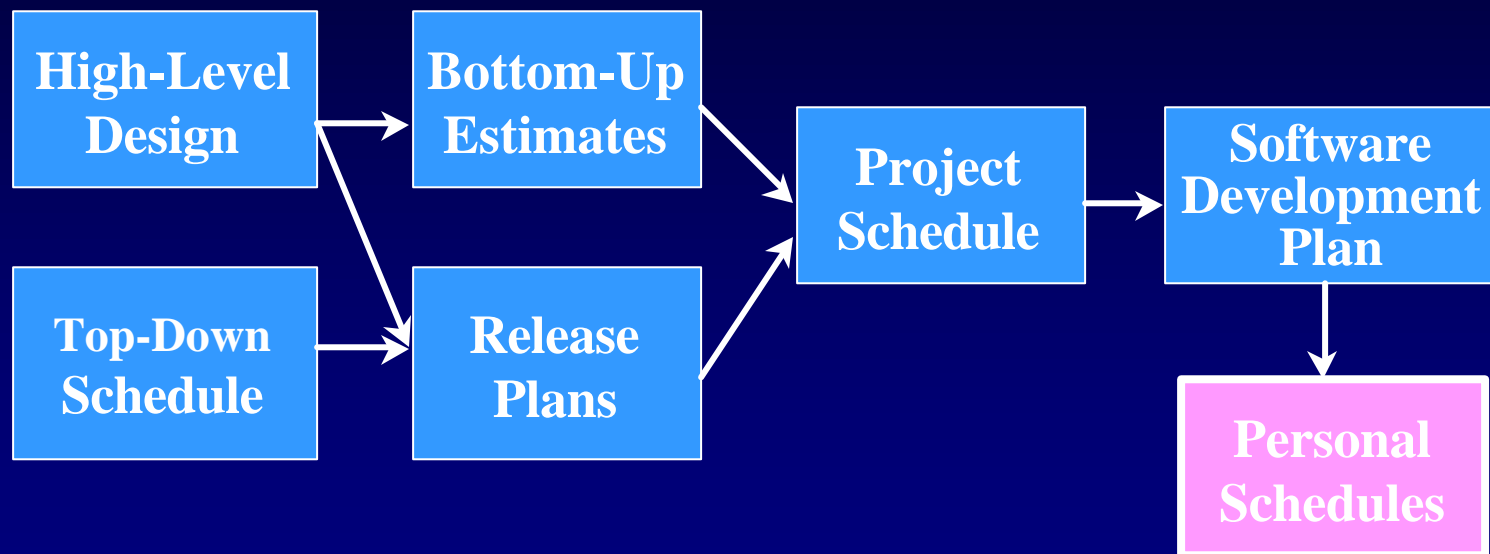
# Software Development Plan

**Summarizes when, how, and with whom the software will be developed.**

**Includes: schedules, engineering release definitions, staffing requirements, subcontractor utilization, project organization, cost estimates, development tools and procedures, task assignments, hardware platform, testing approach.**

**For each incremental release, distribute a “proposed” version for review, and then a “committed” version.**

# Overview of the Process



# Personal Schedules

**Each team member generates a personal schedule for their work on their components for the next release using the software development plan.**

**The personal schedules contain weekly milestones that the project manager can monitor each week and make mid-course corrections, when necessary.**

**Don't forget about holidays, vacations, training, and business trips.**

# Estimation Process Rules of Thumb

<b>High-Level Design Duration</b>	<b>&lt; 3 months</b>
<b>Design Team Size</b>	<b>&lt; 6 architects</b>
<b>Number of Components to be Estimated</b>	<b>&lt; 150</b>
<b>“Paper Design” Time per Component</b>	<b>&lt; 4 hours</b>
<b>Bottom-Up Estimate Duration</b>	<b>&lt; 3 weeks</b>
<b>Time between Engineering Releases</b>	<b>&lt; 8 weeks</b>
<b>Schedule Deviation</b>	<b>&lt; 20%</b>
<b>Overall Project Effort Application:</b>	
<b>Design</b>	<b>40%</b>
<b>Coding</b>	<b>20%</b>
<b>Testing</b>	<b>40%</b>
<b>Milestone Tracking</b>	<b>every week</b>
<b>Schedule Update &amp; Risk Assessment</b>	<b>every month</b>

# Summary

**Schedule/cost planning and design are dependent on each other.**

**The benefits of this approach include:**

- **early performance input**
- **team communication and buy-in**
- **objective data to justify decisions**

## References

- R. Austin and D. Paulish, "A Survey of Commonly Applied Methods for Software Process Improvement," CMU/SEI-93-TR-27, Carnegie Mellon University, 1993.
- B. Boehm, *Software Engineering Economics*, 1981, Prentice-Hall, Englewood Cliffs, NJ.
- C. Hofmeister, R. Nord, and D. Soni, *Applied Software Architecture*, 2000, Addison-Wesley, Reading, MA.
- C. Jones, *Applied Software Measurement*, 1991, McGraw-Hill, New York.
- K.H. Moeller and D.J. Paulish, *Software Metrics*, 1993, IEEE Press, New York.
- D.J. Paulish, R.L. Nord, and D. Soni. "Experience with Architecture-Centered Software Project Planning," In *Proceedings of the Second International Software Architecture Workshop*, 1996.
- D.J. Paulish, *Architecture-Centric Software Project Management*, TBD 2001, Addison-Wesley, Reading, MA.
- D. Soni, R.L. Nord, and C. Hofmeister. "Software Architecture in Industrial Applications," In *Proceedings of the 17<sup>th</sup> International Conference on Software Engineering*, 1995.

# Acronyms

**ACSPP - Architecture Centered Software Project Planning**

**CRS - Component Release Specification**

**FRS - Feature Release Specification**

**HLDD - High Level Design Document**

**SDP - Software Development Plan**

**SIEMENS**

---

## For More Information

**Robert Nord**  
**Software Engineering Institute**  
**Carnegie Mellon University**  
**4500 Fifth Avenue**  
**Pittsburgh, PA 15213**  
**rn@sei.cmu.edu**

**Dan Paulish**  
**Software Engineering Department**  
**Siemens Corporate Research**  
**755 College Road East**  
**Princeton, NJ 08540**  
**dpaulish@scr.siemens.com**