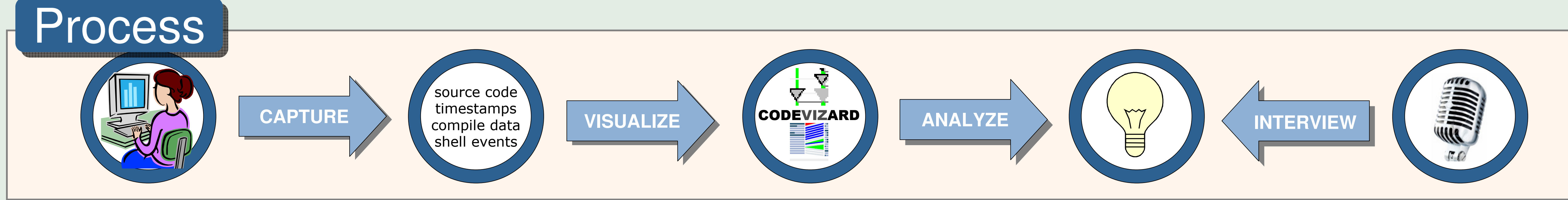


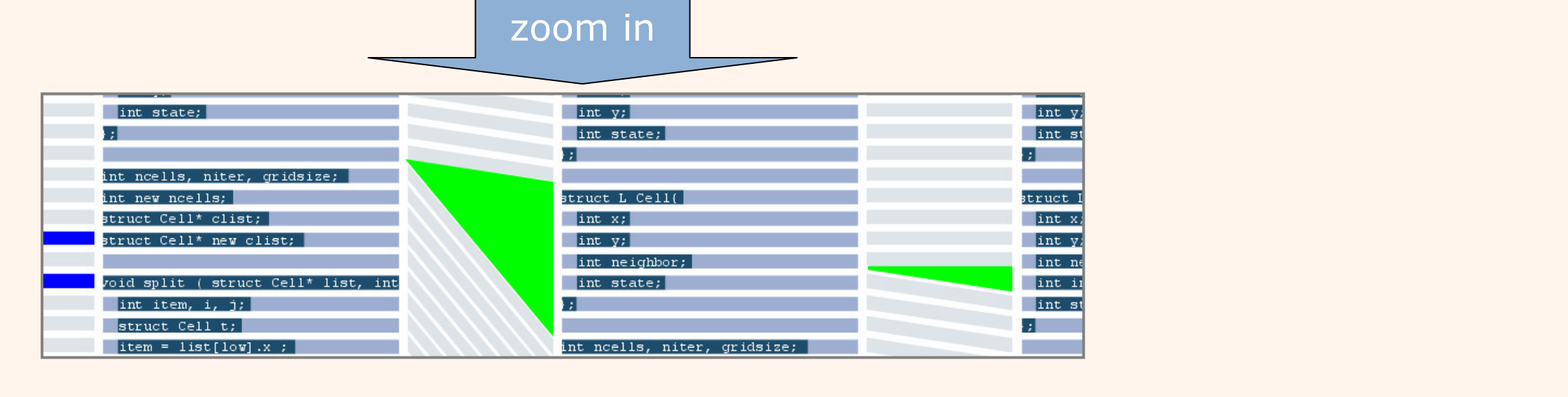
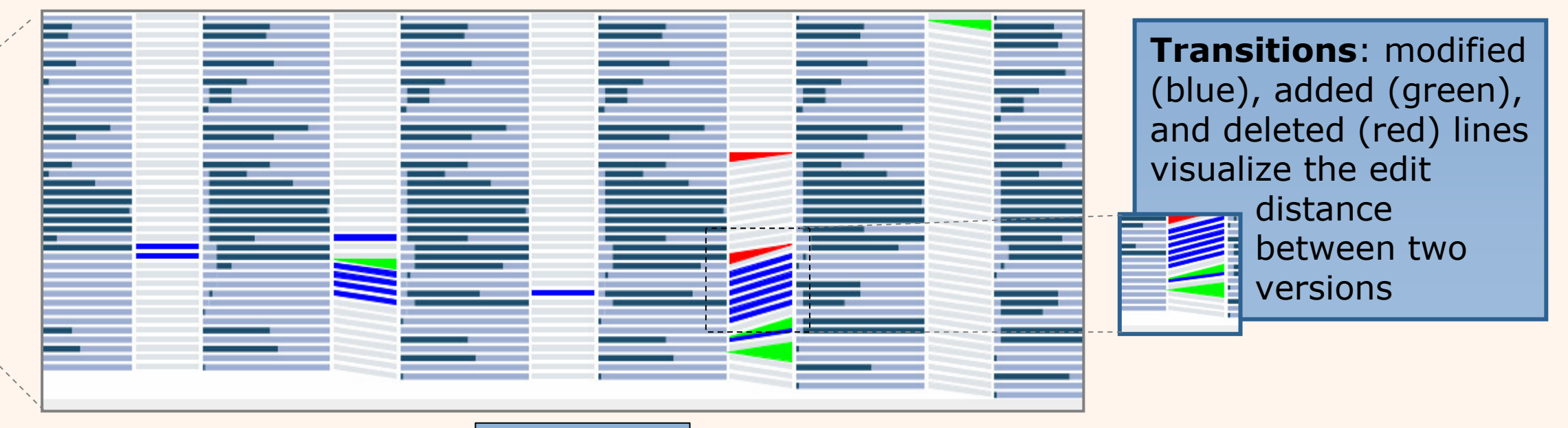
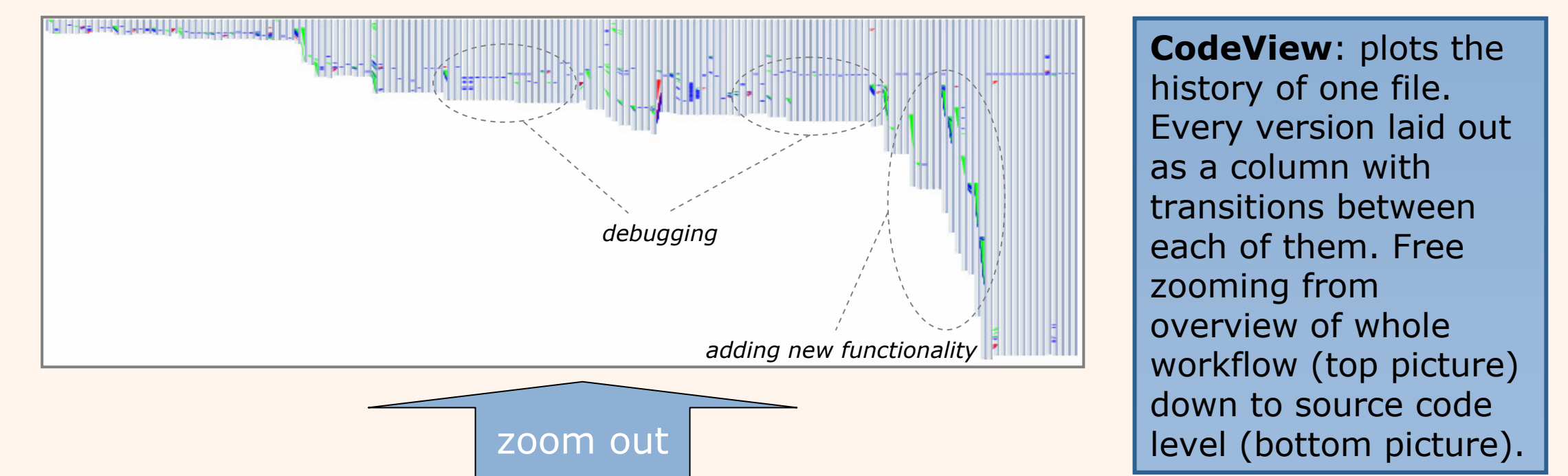
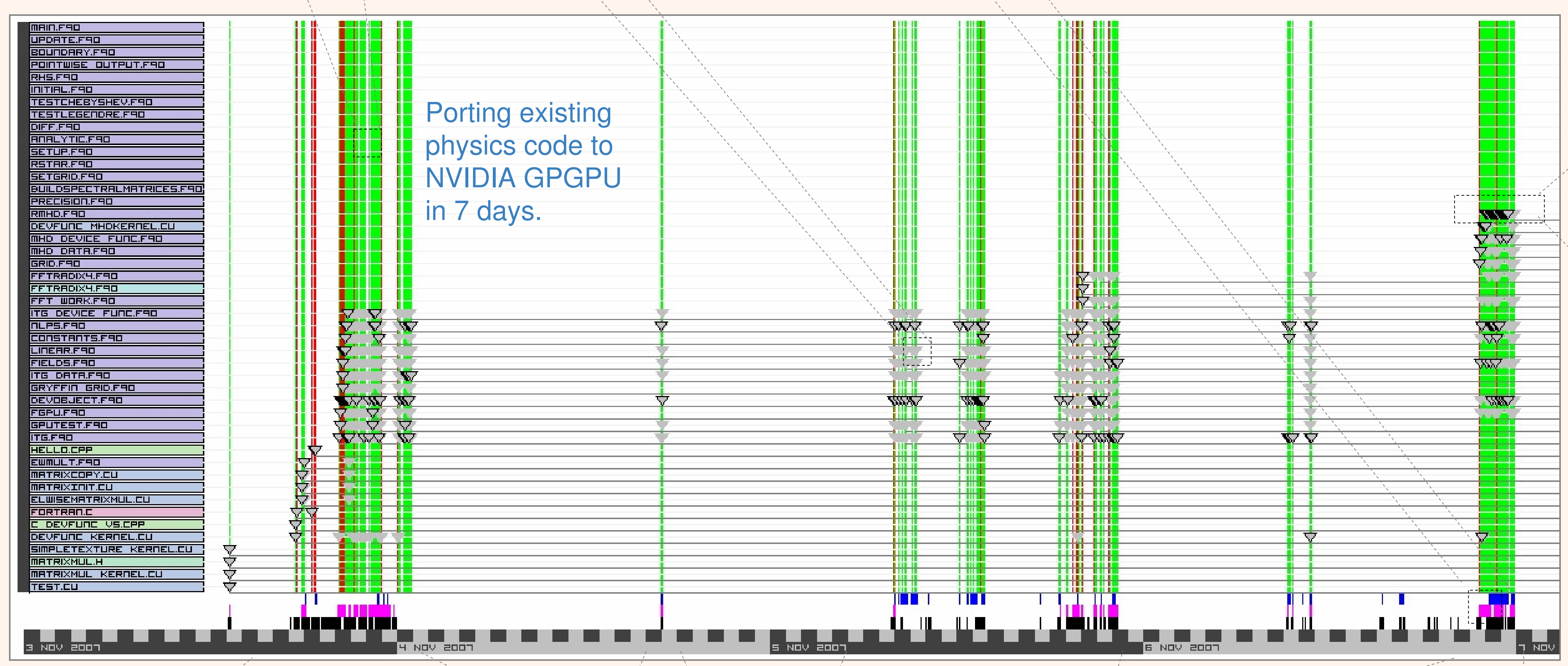
Goal

Identify activities and obstacles in programmer's workflow to *characterize* effort, defects, and productivity.



Visualization: CODEVIZARD

- Y-axis:** folders and files colored by file type
- Compiles:** green lines for successful and red for failed compiles
- File versions with lifelines:** captured at compile time. Black borders indicate that the file has been changed compared to the previous version. Lifelines begins with first compile.
- Shell events:** runs (blue), make (magenta), and others (black)
- X-axis:** time line with hours in first and days in second row



Observation	Lots of failed compiles	Almost no runs	Small changes	Test files	Only one compile 2nd day	3 work-sessions	In first two: no makes, but runs	In last phase new files	High work density	New files, focus on one	compiles makes and runs
Hypothesis	Problems with compiling / running code				Break	First two phases: trying something new Third phase: getting first runs / earlier problems solved			Adding new component, dense and successful work points to error free development		
Interview	Subject used a library and a template code to get familiar with compiling but ran into problems with new environment.				Subject took a one day break.	After meetings with colleagues he got the template code to run in the third phase. He still had to make some adjustments.			The subject ported his code to GPU in short time.		

Results

- We were able to capture and reconstruct the workflow of a seven day development process without interfering with the programmer.
- The data shows that the core work of porting an existing piece of code to a new architecture took less than three hours on the fourth day.
- For the first three days the study participant struggled with understanding the environment (e.g. how to use a library and setting up the right compiler)
- Visual analysis and interviews complement each other.
- Website: <http://www.CODEVIZARD.net>