About Systems Diagrams - The Development System for CSE403

People, Processes and Tools

Looking at "the development system" creates opportunities to change the game. Think of development as a game, with rules, relationships, and so on. What's in the game, and what isn't? What's working? What does success look like? Even if nothing else can change, can we change how we go about doing what we're doing?

> ***Development methodologies, tools, and techniques are instances of attempting to change the game of development. Perhaps unconsciously, the perpetrators were looking to adjust*** **The Development System.**

An Exercise

Do a system diagram (also known as a diagram of effects) about software design. Use the design work your team is doing as an example. Work individually (this time). After you have absorbed the material on how to do a systems diagram. (References provided separately) spend 15 minutes tops, doing the actual work. Try for 10.

As you diagram the influences around doing design in your project, you will notice influences that you're not sure about. Does this thing help, or not? Does more of this thing mean more design, or less. Those are the useful questions.

Techniques for Making Systems Diagrams

The problem with doing system diagrams is starting and stopping. Here are two starting tricks:

- Tell a story. The nouns are things. Turn each noun into a measure – something that can be counted. The verbs in the story are changes in things, which have to be explained in terms of an "influence". Like this:
    - Today our team sat down with to make a design
    - Hmmmmm. Make. OK, how does that happen? From what? From requirements.
    - So, more requirements means more design. Is that so? What's that have to do with code – more code or less code?
    - Etc.
- Start with an artifact. So, we make designs. Now the design comes out quickly when there aren't a lot of requirements, but slowly when there are. And when the design uses familiar tools, or addresses a familiar problem domain, it goes faster. So, there are some influences there. . .

Because everything is connected to everything else stopping is also hard, once you start. You need a halting condition to doing a systems diagram. Here are four of them:

- Time box it. Draw measures and influences until the time is up;
- Use an optimization strategy – drop out the small effects, or infrequent influences. When you've got 90% of the measure accounted for, stop.
- Stop when it explains something new. Oh, so that's why we're suddenly late – more requirements means more design work.
- Stop when you get three questions about what influences what. Three is a good number of questions to play with at one time.

How is a system diagram different from a use case?

A use case is procedural, and / or represents a sequence in time. This happens then that happens, then the other happens.

A system diagram is a generalization that exists outside of time, showing relationships. In that sense it's of the same family as an E/R diagram or a class diagram – closer to the former. A use case would zip around in the diagram of effects as you step through it. In fact, that's one way to elaborate a diagram of effects, if you've got some use cases.

A system diagram shows relationships between measured quantities. In principle there's a process there to make the quantities relate. Gas in the tank has a positive relationship with money spent on gas, and a negative one with miles driven. That's more abstract than the "get gas" use case, the "go someplace" use case, the "ran out of gas" use case, etc.

For greater clarity on this distinction, investigate "aggregates" in either of Weinberg's books on general systems.

When to use system diagrams (diagram of effects.)

A System Diagram is most effective when you're puzzled, often about a cause. Something strange happened, and you don't know why. Something's not working, and you don't know why.

- When something weird happens that you can't find a reason for, a system diagram can uncover a hidden influence. One special case of a hidden influence is a hidden coupling – an influence between two measures that are already on the diagram, but have an interaction you haven't noted.

- When the law of unintended consequences bites you in the butt, and your grand scheme for making things better backfires, a systems diagram can help figure out what happened.

- When you've got a desired outcome, and don't know how to get it. "We want more design stuff, faster." OK, what influences the amount of design you get? More designers? Not always.

- So, if you're not getting enough "stuff" fast enough in developing your system, a "systems diagram" of your environment – your development system – or part of it, can help you find things to change.

There's a lot of "management" here – what's that about? Well, the people generally responsible for performance and productivity of a collection of resources are generally called "managers." When you change the development system, you're changing the resource mix, and the performance point of that system. That's a management thing, at least in part. So, the people who think deeply and discuss how development works are often the people concerned with managing development.

Another Exercise, for a Team

Break into teams, of at least three, but no more than six or so. If you have separate development teams for various components, select the teams for this exercise across the development teams. Get yourselves some space, a flip-chart, and other materials for capturing ideas.

Work in your team:

1 - Name the top three annoying things about the development system you are in that you would like to change. 5 minutes, max.

2 - Develop the top, most annoying thing, with a systems diagram. (10 min).

3 – Identify at least one intervention that might address the problem. Put that on the system diagram. (5 min).

Then each team in turn will brief back to all of us at 5 min each:

1 - Name your three issues.

2 - Show us the system diagram of the problem.

3 - And the candidate solution

Discuss as a whole (10 min):

1 - What's similar about the problems?

2 - What's different?

3 - Identify uses of common tricks for solving systems problems:

- Change the boundaries.
- Remove a coupling.
- Just work harder.

Now, let's all pick one problem, the one with the most #1s. Then pick one solution, from the ones suggested. Break up into your groups and spend 10 minutes:

1 – Find three ways the intervention might go wrong

2 – Describe these on an expanded systems diagram.

As a group, what did we discover?

Key observations:

- Development is a system.
- You're in the system.
- So is everybody else. (Which is why this touchy-feely people stuff is such a big deal™. Performance in the system, and change to the system is all about people.)