Detta är ett av de mest grundläggande mindseten och metodikerna, eller snarare områdena inom digitla produktutveckling och rör systemteori eller systemics.

Det finns så många bra beskrivningar av vad systemteori är så det går vi inte in på här. Istället adreesserar vi en egenskap hos system och det är dess oförutsägbarhet och att system har en tendens att yttra sig o-intuitivt vilket är vad vi framför allt vill använda detta till i detta arbete.

Ramverk som Edgy visar att allt är ett system på ett tydligare sätt än Archimate eller Zachman. De senare beskriver också vad systemet består av, men mera bara som en ontologi och utifrån hur saker hänger ihop, hierarkiskt eller konceptuellt. Edgy gör det också, men dess koncept med intersections och beroendepilarna som går mellan elementen, är mer sann utifrån systemteori.

Edgy visar t.ex att om man ändrar i Product så påverkas Users experience eller Organisation och från det kan man förstå att man måste åtminstone beakta vad denna påverkan kan vara. Direkt, indirekt, dämpande/förstärkande via olika starka feedbackloopar osv.

Men Edgy visar också att saker är en hieraki, att Architecture består av olika element osv. Så det finns med, liksom det gör i Archimate.

Det pattern vi kallar för ’Pigalle’ är också en slags systemics med ett system bestående av två fundamentala delar, en insikt och en produkt som adresserar denna insikt. Traditionell utveckling går via faserna insikt, analys, design, implement och product, en slags ’push right’. Pigalle implementerar designtänkande genom att vända på denna kadens och börja med produkten och genom den via lärande ge samma resultat som analys, design, implementation gör. Detta är systemteori praktiskt genom att säga att insikt och produkt hänger samman och går lika bra att utföra 'pull left’ bara man använder rätt arbetssätt/way of working.

Angränsande pattern är ett av de andra vi redovisar, ’Indigo’ med dess fyra eller fem(Problem) eller sex(Research). Det systemiska utgörs av att man kan researcha o analysera kring behov t.ex, sedan design. Det är tradtionellt. Men man kan också, systemics way, gå andra vägen via Product o lära sig den vägen. Det viktiga är att förstå att Product också påverkar Needs, Design, Problem om man hanterar relationen i systemet genom thoughtful experiments. Dvs sätter upp vad man antar o hur man kan lära sig utifrån experiment.

Det vi vill lyfta med praktiska leverabler i form av metodiker bygger alltså på detta grund-pattern som kanske aldrig får en egen metodik eller verktyg utan bara agerar indirekt.