# Circle Language Spec Revamp Notes 2019-08

*Author: JJ van Zon*

*Location: Oosterhout, The Netherlands*

*Date: August, 2019 – April 2020*

## Contents

[Contents 1](#_Toc37540046)

[Project Outline 2](#_Toc37540047)

[Introduction 2](#_Toc37540048)

[Goals 2](#_Toc37540049)

[Limitations 2](#_Toc37540050)

[Steps 3](#_Toc37540051)

[Requirements 3](#_Toc37540052)

[Information 3](#_Toc37540053)

[Notes 3](#_Toc37540054)

[2020-04-01 Notes Circle Language Spec Planning Docs 3](#_Toc37540055)

[2020-04-09 Brainstorm Explore (Content) Search Options 6](#_Toc37540056)

[2020-03-22 Notes Circle 3 Programming Planning Docs 6](#_Toc37540057)

[2020-03-07 Brainstorm Circle 3 Programming Planning Docs 6](#_Toc37540058)

[2019-12-29 Brainstorm Scope 7](#_Toc37540059)

[2019-12-29 Brainstorm Circle Language Spec: Design 8](#_Toc37540060)

[2019-08-05 Brainstorm Circle Language Spec: Writing Style 9](#_Toc37540061)

[2019-08-05 Brainstorm Circle Language Spec: Large Lists Problem 10](#_Toc37540062)

[2020-01-13 Brainstorm Aspects / Concepts 10](#_Toc37540063)

[2019-08-05 Brainstorm Version Control and Open Source 10](#_Toc37540064)

[Done 11](#_Toc37540065)

[2020-04-01 Done Circle Language Spec Planning Docs 11](#_Toc37540066)

[2020-03-22 Done Project Names 12](#_Toc37540067)

[2020-03-22 Done Organize Planning Docs 12](#_Toc37540068)

[2020-03-16 Done Circle 3 Requirements Docs 13](#_Toc37540069)

[2020-03-15 Done 14](#_Toc37540070)

[2020-03-08 Done Reading Circle 3 Requirements Docs 14](#_Toc37540071)

[2020-03-07 Done No Planning or Docs Back Then 14](#_Toc37540072)

[2019-12-29 Done Brainstorm Scoping 14](#_Toc37540073)

[2020-02-23 Done Reading Circle 3 Strategy 16](#_Toc37540074)

[2020-02-20 Done 16](#_Toc37540075)

[2020-02-16 Done Notes 16](#_Toc37540076)

[2020-02-13 Done Notes 16](#_Toc37540077)

[2020-01-13 Done Notes 16](#_Toc37540078)

[2020-01-30 Done Notes 17](#_Toc37540079)

[2020-01-04 Done Rough Plan 17](#_Toc37540080)

[2019-12-15 Done 17](#_Toc37540081)

[2019-08-10 Done Brainstorm complexity in preserving rename history 17](#_Toc37540082)

[2019-08-11 Done Basic Math Conversion to Source Control History? 18](#_Toc37540083)

[2019-08-11 Done Controls Concepts Conversion to Source Control History 18](#_Toc37540084)

[2019-08-05 Done Brainstorm Restructuring Docs 18](#_Toc37540085)

[2019-08-04 Done Notes converting version folders to source control history 19](#_Toc37540086)

## Project Outline

### Introduction

Circle is an unfinished programming language.

I spent thousands of hours spread out over a few decades with this idea of how to visually express the internals of computers and programming languages.

Circle Language Spec is an unfinished programming language design. The same folder of docs contain texts about all sorts of software applications. But this project 'Circle Language Spec Revamp' is really about the programming language specification.

### Goals

#### Main goals

* 'Soft' goals:
  + Work on project that seems to matter.
  + Keep my head occupied with something else.
* 'Hard' goals:
  + Isolate the Circle Language docs from the rest of the docs.
  + Open source the Circle Language docs.
* Low priority:
  + Get the documents in a state better accessible to others.
  + / Make the documents usable for implementation.
  + / Make a technical design

/ To highlight the challenges of how one might implement this as a working piece of software.

### Limitations

* Do not program.
* Limit it to the 'Language' part of the docs (not 'Framework', 'Operating System' or 'Applications & Media')
* Do not open source the prototype apps.
* Before I start changing anything, I might just read the material as-is.

(I have great difficulty maintaining this rule. I tend to immediately want to start changing things.)

* I might want to focus on splitting apart non-Circle-Language topics from the Circle Language topics.
* Maybe not write to clarify new concepts, but rather limit it organizing already worked out concepts.

### Steps

* Reorganize
  + Convert docs version folders to source control history.
  + Convert project docs version folders to source control history.
  + Categorize loose ideas in in Ideas.doc, so Circle Language and Circle 3 Programming are isolated.
* Reorient
  + Read project docs sequentially for thorough orientation.
* Improve planning docs:
  + Reorganize project docs
  + Remove studying goals.
  + Split documentation goals from programming goals.
  + Less resolute language, more wiggle room.
    - (Perhaps search for definites such as 'have to', 'must', 'will', 'should', 'very'.)
    - (The 'Productive Writing' part of the Strategy doc is quite resolute in the language.)
    - (Maybe I am being too strict towards myself, about having to rewrite everything in a milder tone.)
  + Scoping
  + Include hour sheet?
* Improve Circle Language Spec:
  + Reorganize Circle Docs files and folders (e.g. lone files in folders is not handy).
  + Scoping
  + Turn separate Concept/Diagram/Text Code articles into one.
  + Reduce the number of articles, by grouping topics together.
  + Generally go through content reformulating things?
  + Less resolute language, more wiggle room?
  + Add pictures to the 'walls of text'?
  + Start reading / writing / reformulating.
  + Convert from Word to markdown, so it may get indexed by Google and easier to navigate and read once published.
* Separate git repositories:
  + Split off Project Docs for Circle language documentation into a separate git repository.
  + Split off Circle Docs' language specification part into a separate git repository.

### Language Design / Content Changes

* Gentle intro
* Static notation
* Large lists problem
* Friend notation (too prominent?)
* Fully OO (is it?)
* Language lost its purity (with implicit notations?)
* Comment notation
* List concept more generic
* Pointer to pointer (move to separate chapter?)
* Relationships 'always' bidirectional (might not be a good choice)
* Object-commands / command-objects exchangeability doubts
* Enriching misc diagram topics
* Writing style:
  + More pictures
  + More open, less resolute language
  + Use common IT terms
  + Write unfinished articles
  + Is it a *notation* or is it a *run-time*?

## Information

* For converting to Markdown:
  + <https://pandoc.org/index.html>
  + [https://word2md.com](https://word2md.com/)

## TODO

### 2020-05-24 Notes Planning

* Maybe merge more article together in the first chapters.
* Maybe list out main language design issues.
* Some work in Future Sub-Projects may already be done.

### 2020-04-18 Brainstorm Writing Style

* The aim is to use less definite, more open language and to take out the personal 'I' form. Consider reformulating command-like sentences and rule-like statements to add maybe's and perhapses or past tense instead of imperative tense. Avoiding the word 'you' also seems to let the air in (it feels less like a chore). Even the word 'is' can be an expression of something being definite, which you may want to loosen up a bit.
* Also check: whether the stories make sense.
* And whether the content is still in scope.
* Remove links, since they break so easily.
* That seems quite a lot to check. Maybe that is why it is not easy.
* Casual mentionings of article titles are also fragile. They break quite easily.

### 2019-08-05 Brainstorm Writing Style

* The read uses terminology in a very specific way, that is not shared with my peers, therefor not easing readers into the material. Maybe an introduction would do. The 'older' versions actually seemed to give that more gradual intro, that takes the reader by the hand, leading them to how and why this might actually be useful, rather than just plainly stating the shapes that the diagrams can contain.
* That said, I do want to quickly come to the raw definition of how the diagrams are built-up. To introduce the raw basic specs quickly.
* The work might be modularized. I am not attacted to how CSS3 is modularized, each piece of specs with a different state of being finished up. It seems messy. But I can employ the same organization to accept certain concepts are just more crystalized out than others, making it easier to share, even in an unfinished state, and stimulating keeping things separated and separately usable even when other parts are just really still messy. > Scoped things more sharply instead, so more is finished up better percentually.
* I seem to have had several goals fighting over each other, in projects done long ago about this documentation:
  + Explaining it to myself.
  + Designing the concepts, separately from the notation.
  + Tying together loose ends.
  + Making it easy to read for someone else.
* Maybe the design of the programming language should lose some ambition and express that only as dreams.
* I took a look at some of the postponed work. I worry about the messiness of the content. And if the loose ends will make the idea fall apart. And whether this makes it even fit for publishing. I just don't know. But I think I should come back to it later. Because I had strategies for this. And I might be too hard on myself. A clear 'flag' [Preliminary documentation] in red somewhere at the top usually does the trick. Might tells people clearly not to not too much from the text that follows. Just being clear about that might be enough.
* Reading over the Interfaces planning docs I noticed I said there I like how the Interfaces chapter reads. I started reading it over. I notice, I only like how it reads where lots of little diagrams are shown. I like it not so much when I just see walls of texts. I think it is my 'brain type' so to say. But I think having both text and pictures would help a lot of brain types. So: more pictures. I like pictures.
* I notice I talk a lot about implementation rather than notation. For instance: Does a dashed circle mean it is used as a class, enforced to be a class, static inside its container, how does it work in the system interfaces? What if it is just the notation that is a good idea, what if the implementation isn't. That might even make system interfaces' precise definition not important or maybe just subjected to diagrams drawn out to represent things from another language, like C#. C# getters and setters might be in a system interface notation. But setting an object reference's interface dynamically in runtime… may be too much of an implementation detail. I think it is a language definition / runtime implementation separation. In think the engineers at Microsoft might be right about developing language spec / runtime / framework / compiler quasi-independently. Maybe I can be inspired by that and make my language definition a litte simpler. I am subjected to the pitfall of wanting to cover every little minor edge-case, of which I have a fear that it may make the whole system fall to pieces. I already warned myself about that in the Circle Language Spec Strategy document. But now I think other people might actually read this, I start to think: maybe limit the scope. Somehow define the diagram notation and what it represents and not want to work out how things would work in a runtime. Runtime would be a system where the diagrams and actually the data that internally describes the diagram, to be loaded and run as computer programs. I think I wanted to check the usability of the notation by shining light on any little aspect of it, I could find. But I think some details are not that important. Maybe those are to be demoted to possible implementation details, to keep the main part of the story clean. I am OK with apologizing in the documentation, that this might not be usable or something. The description in the Strategy document is pretty much spot on, I think.
* I get the problem that next to introducing new notation, I also wanted to introduce new concepts. A new conceptual take on things. I think it all became a little much. Maybe I can improve that during this project 'Circle Language Spec Revamp'.
* I think the 'being blunt' might help. It is not blunt unfriendly, because the explanations might become so much simpler if you say: this is that, this is that, instead of and this far-fetched edge case is solved in this difficult, abstract, theoretical, but precisely determined way, that I'm not sure I even understand anymore… : )

### 2020-05-12 Brainstorm Language Design: Reservations about 'Fully Describes OO'

* Here and there I mention that the diagram notation would 'fully describes' an OO system. But I do not mention that it stops at arithmetic, comparative, logical operators and such. You might display that as an object graph. An idea was that 'Math' would be an external module that describes all the basic operators, but another idea was for expressions like that could be just included as bits of text code inside the diagram. Maybe not be too strict about this towards myself.
* Also when I look at Circle Construct Drafts, I see topics one might expect in a language spec, that are not addressed. I guess it does not have to have everything any other language spec might have…

### 2019-12-29 Brainstorm Language Design & Content Changes

* / Even the large 'introduction' document to 'Software System': you might put part of it in one for 'language' and just refer to it from the main document.
* Rename the term 'Code Base' to something like 'Base Code' or 'Base of the Code'.
* I think that somewhere along the way, the language lost its purity. Hypothetically, I may have gotten carried away a few times. For instance, using the dashed line as a conceptual expression of the idea of 'classes' or 'types': I think I tend to introduce ideas about notations that might simplify things visually, but possibly introducing ambiguity. At first, the language was intended to be the purest form that I could find, in which you could draw out an object oriented system with shapes and lines. What happened to that as I started to make drawing something with dashed lines something ambiguous? I am not sure: this might be a non-issue. But maybe I want to be wary of where I got carried away and not think in definites about the final form of the language. This also counts for e.g. System Command Call with Argument Notation: I introduce simplified, yet abused slightly ambiguous notation for something you probably will never see used. What if at some point I find use case for this simplified notation that also looks like something it's not? E.g. a call to an object getter:



* It's ambiguous, because it looks like the diamond, which is a command call to the getter, is a reference to the object. It reference-associates a command call and an object, which are two separate 'objects', but connecting them with solid lines would suggest they are the same object.
* Maybe expressing doubt about it at the beginning is just as far as the solution might go.
* I am missing the notation for comment. Would be welcome. I think it is simple, unambiguous and would be a shame if it were overlooked as an option. Just a line from the text to the wherever, as long as nothing's directly connected to it, there should be nothing else that would clash with that symbolism.
* The protected access mark / access modifier:



I thought of this relatively early. To solve ambiguity problems, I introduced notations such as: a system command connector for the New command:



Event connector:



The concepts of its being *required* for a command to be executed:



But the latter notations are there kind of because the simpler notation was already in use by the concept of the protected access modifier.

I am having second thoughts about this. Maybe I would like to consider using a different symbol for Protected, so that these connectors, events and optional/required can get a less contrived notation.

Another example where it is an issue: This expresses that the other end of the connection should be a triangle:



But if you would accidentally bend the half triangle at the end the wrong way, it would look like a protected access modifier. Well… maybe that is not definitive proof that there's an issue, but it's an indication of something might be confusing.

* A jagged line type for the Name aspect I would like.
* Maybe pointer-to-pointer situations might all be moved away from each chapter and into a separate chapter. It seems a specific issue and many things seem to be able to live without it.

### 2019-08-05 Brainstorm Language Design: Large Lists Problem

* One point of failure I see in this computer language, is that it works well when there are a limited number of symbols, but as lists get big, the language seems to lose its effectivity. One way to still make it useful, is perhaps to filter, or only partially display lists in the diagrams, just like a normal grid or list would. There was a prototype app that would generate diagrams out of vast sources of symbols. The problem became apparent there and it has been in the back of my head since then.

The 'large lists' problem in Circle also applies to large lists of commands, that might apply to an object.

Also that UI's are often optimized to show the most relevant options and then I (with poor judgement?) just say: no none of that, everything only.

Maybe it's just that this UI will have its place along side other techniques.

It's just that the large lists problem should be solved in my view.

Maybe permanent filtering and reordering, which is often hard to customize in windows programs. Like a menu customizer.

Try to make it easier to do that. Construct your own limited view.

### 2020-04-19 Brainstorm Conversion to MD

* I thought MD would be the way to go with this project compared to docx. But I have doubts if MD is the better way to go. I keep getting stressed out about those doubts. It makes my decisions on what to do with the documents (formatting) harder, because I just am so unsure about this. What I would like is to have a good MD editor. I just hope some of the difficulties editing MD compared to docx can be taken away by a good tool. Also: I think MD will be vieweable and better indexeable on the internet, but I have not really thoroughly seen that confirmed. The main problems with MD editing I have is navigating headings and sections around and viewing the outcome of my formattings better as I type the MD in. The doubts about indexing: I know GitHub can show the MD in nice formatting, but will a lot of MD's actually be nicely navigatable? Will those MD's actually be indexed, or does that only count for e.g. the REAME.MD in the root dir of a git repository? another doubt about MD is: can I be satisfied with some of the tools that convert docx to MD? Will that all be doable? It's so many articles. Can I do it in bulk? Will I merge more articles into one making converting one by one by hand better. How does it work with links but especially with images? All those doubts made me postpone decisions about converting things to MD, but that makes me burdened with formatting docx's making them look nice, which is not that useful if I am going to convert all of that to MD anyway. That last argument does not seem very strong to me. If I know that MD indexes well, that might already make some decisions to go for MD easier, when I decide e.g. about small documents.
* So with all those doubts, I want to explore MD options a little.
* Main points:
  + MD editor
  + Will MD show and index better on the internet?
* Notes:
  + Trying to research this puts a strain on me.
  + I found out relative links to images work nicely in MD on github.
  + I cannot find if google will index the content well.
  + I do know that clicking an MD in GitHub leads to nicely readable page. That's something you cannot say about docx's.
  + That last point may actually convince me already to make MD's more often.
* MD editors:
  + Visual Studio code has a preview view and outline view: <https://code.visualstudio.com/Docs/languages/markdown>
  + <https://duckduckgo.com/?q=MD+editor&t=opera&ia=web>
  + <https://marketplace.visualstudio.com/items?itemName=MadsKristensen.MarkdownEditor>
  + There seem to be options.
* Alternative: Convert the Future Sub-Project Summaries to MD.
  + Pro: Those future docs are short and don't need much formatting.
  + Con: MD is harder to edit (?) > Needs research.
  + Pro: MD is more accessible on the internet (?) > Needs test.
  + I would like to have the arguments for and against better proven.
  + But it is also not directly my goal right now.
  + It would help me to know if MD really has those advantanges. Otherwise I would keep being confused about whether or not I should convert to MD.
  + But pragmatically: That was not what I was doing. It was formatting the docx's, what I was doing. But pragmatism is sometimes something I feel others have determined for me. Maybe knowing something for sure is more pragmatic for me, because the doubt causes me so much stress.
* Concession: Do both. First format the 4 remaining Summary's. Then research MD more.
* Conversion programs:
  + Pandoc: I did not like it. It was expecting a user interface, but I got a command line interface. Also, the output from docx to MD is not what I expected. I got quote markup > for just indentation and a dashed bulleted list got the dashes escaped or something \-. The check list style for GitHub that's kind a sorta already MD format, got escaped it seemed. I get not everything is unambiguously translatable, but this was too far removed of what I would type in myself. So sorry, pandoc. Moving on.
  + I might as well just copy paste to a text document and then do it myself.
  + But I am going into reseach again, and that is not what I want to deeply into. I do not have the energy.

### 2019-08-05 Brainstorm Version Control and Open Source

Is it really such a good plan to open source it? Am I really not just throwing away money? Maybe it is better to wait with that decision, after I know where things are heading with work and income?

There is an '… IPC Parse.doc' text in the source control history. It should be deleted. I guess I have to search for intellectual property problems. This in docs that are so intellectual property sensitive. Before open sourcing it, you need to do this intellectual property check.

There is a folder 'Previous Versions' that would make you think you forgot to put those at the beginning of the conversion from version folders source control history. But they are not necessarily previous versions of documentation, but more like previous versions of systems.

2004-00-00 XX Symbol Language\Symbol Pictures (Keep Packed, Paths Too Long).zip has intellectual property problems in Pictures\Diagram Examples.

How do you even license documentation? I wanted to throw an MIT license against it, but that is for code… hmm…

<https://dreamsongs.com/IHE/IHE-50.html>

My brother's first name is in some sub-project planning doc. I can take it out of the current version. I should ask him if he is OK with it staying in the source control history or if he wants it out of there.

Is the summary of the C and C++ in Summaries of Other Languages an intellectual property problem? I summarized the languages using MSDN as source information. Hmm… If I didn't mention the source information it would not look like an intellectual property problem. Just I read stuff about a programming language and wrote stuff down… not sure.

### 2019-12-29 Brainstorm Scope

* Brainstorm is done, but useful for reference.
* Out of scope:
  + 'Operating system' like components
  + Studying
  + Fundamental principles
    - Basically all of them are probably out-of-scope.
    - Do I limit the ambitions with the project?
    - Do I remove fundamental principles that are questionable or irrelevant, like that the code base is written in C++, or things that may speed up development, things others could figure out, making how I feel about it not add much? Maybe ease up on things, talk less strict about things?
  + 'Software System': an abstraction layer above the new computer language.
    - I think I should take interesting parts from the general Software System documentation.
    - Then drop them somewhere in the Circle Language Spec documentation.
    - Also for the planning docs: there are Software System Documentation out-of-scope projects. Perhaps part of it was Circle Language Design.
    - > Not much usable content is still in scope.
  + Computer language topics out-of-scope:
    - Concepts / aspect oriented programming
    - Database principles
    - Concept libraries
    - Machine language
    - Internet as a single computer
    - So many things, but I want to leave them out.
* The data concepts and coding concepts thing, and the aspect oriented-like thing, I may want to put that out of scope. I might want to accept that the idea I present has limited potential, and might not apply to what you can do with a database, or ambitious aspect oriented programming ideas… just object oriented programming expressed in diagrams is enough. I did not have a clear idea back then how to merge the two or three concepts into that diagram language. I had some general ideas, but not entirely concrete. I should just leave it at that. That seems more achievable.

### 2019-12-29 Brainstorm Scope: Diagrams / Constructs / Gap Lifting

* Diagram topics:
  + Eventually, you might split the now in-scope topics apart in pieces that are in-scope and others that are out-of-scope. Some ideas just aren't the original ones, not a hot idea, and not prone to be adopted. For instance: The spaces in identifiers, text code ideas… I might have done good by keeping conceptual thoughts apart from diagram thoughts. I tend to want to merge those two aspects for the benefit of the reader. But I also tend to want a diagram notation that can be applied to other programming languages as a source for the diagrams. In that case, not all wild conceptual ideas are relevant. I did have thoughts: how would I implement this if I did it from the ground up, how would I implement that if I did it from the ground up, like interfaces, inheritance, ref-ness etc.
* Gap lifting:
  + Also I had ideas on the other end of the spectrum, more macroscopically: how far can I take the application of this diagram notation? Could they even substitute user-interfaces? Could boundaries be lifted, for instance, between things running on different computers and the boundary between users and programmers. The language gaps between things like files and folders, databases, programming languages, user interfaces… gaps between the physical and the logical. All that. I thought that was interesting to think about.
* The application of them are different (of *diagrams, conceptual constructs* and *boundaries lifting*).
* Proposed constructs:
  + Another example is the automatic diagram organization topics. The diagram notation idea can live without some of the wild ideas in that. For instance, interchangeability between containment and referential structure or inversibility of containment in case of bidirectional relationships. Also the striving to want almost all relationships between objects to be bidirectional, does not apply if you want to use the diagrams to express systems in which you have a choice if relationships are bidirectional or unidirectional. Also giving things a different name (aspects are all of a sudden called concepts) is not a priority, and perhaps even alienating. Those are just some ideas I have about how to pull things apart.
* I think maybe those differences in application call for a rigorous split up: *diagram notation / redesigned constructs / gap lifting*. Diagram notation is a bit of a grey area, because it wants to use constructs. Some of the notation capabilities imply different variations of otherwise fixed constructs. Black box is a notation I dislike. It works, but it is not elegant (I think). Also, if you take C# as a source, rules about private/public are set, and some things just don't have to be denoted, because they are implied. A method contained by a a class can access all the other private members of that class. In my efforts to generalize and make things interchangeable, this 'had to be' explicitly denoted… I think a separation of some main groups of concepts is appropriate here, so the ideas are better transferable and perhaps better usable and applicable. It might make the general idea about the diagram notation more pure and also more freely usable. An argument that falls away with this open sourcing compared to patenting is that things do not need to have a closed, unambiguous definition. Things can be just part usable, using and replacing rules as one wishes. You can say at some point the diagram notation surpassed the original goal on two ends. I had somewhat of an explosion of ideas then surrounding this notation. Boundaries between ideas were not well defined. That's a (good?/bad?) quality of my way of thinking, I guess. But I tend to want to focus things, by splitting things apart in the main blocks, so it that it might become more practical (for others or myself).
* "Circle Language Spec" / "Circle Language Broader View" / "Circle Language JJ's Construct Proposals"
* I don't like the last name. It seems long.
* Proposals seems more community-based, not single authorish.
* Language Spec vs Broader View:
  + The idea that the diagram expression should be canonical and unambiguously express anything from any computer language, might not be a rule I want to uphold in the Circle Language Spec doc. For instance if C# has certain rules for scoping of implied accessibility rules of members, you might not want to express that in a diagram. It sort of would make no sense if the diagrams are applied like a helper tool in Visual Studio to visualize certain aspects of C# code, to have symbols that disambiguate something, that is not unambiguous according to C#. The concept of canonicalizing things, may have a better place in Circle Language Broader View docs, I mean to isolate from the Circle Language Spec. A reason for unambiguous expression may only become clear, if you look at those ideas about possible broader applications of this notation: specifically where in a diagram you just switch from one source language to the next, by navigating the symbols. (E.g.: Now the diagram expresses something with C# as the source, navigate onward and you may see some diagram expressing something that came from JavaScript… the rules change, maybe the diagram expression should be unambiguous.) I think it is good and keeps it simpler and ideas less stuck on each other, to speak in options for diagram expression. There is a general theme in the expression. It is not all wishy washy, but there is wiggle room for how to use it. Lots. I just want to share the idea, not impose how it should be used.

## Postponed

### 2020-05-25 Postponed Content Changes for Relations

I question the way the Relations topic is approached.

I think it is an interesting idea. But it seems presented in a way that all relationships are supposed to be bidirectional. In one of my prototype apps for a programming environment this idea was sort of centric. But now that the notation seems more important to me trying to push constructs that might not be that widely applicable, this seems to disturb the story.

It seems to be more about an idea for ubiquitous bidirectional relationships. It seems that Relations in my view back then was synonymous for bi-directional relationships and my idea was you should make most relationships bi-directional. But the problem with, that I have with that now, is that doing so sort of removes hierarchy: everything seems to land on the same level of the containment structure, if every relationship is mutual. I had some ideas how to then solve that again, with relation direction. I think I still like that idea. But now I have a construct proposal, that seems to be pushed too hard to the foreground, that seems something you should be able to choose, not something you should be pushed down your throat, excuse the sarcasm. And unless I finish it, it hinders the notation, that I find more important than having all relationships be bi-directional.

Maybe I might demote the Relations construct to Circle Construct Drafts entirely. Maybe. It seems about what you could do with these unfinished 'Creator' systems I once programmed. But what you could do with Creator had to do with aspects, relationships, libraries of aspects, that tap into a way to describe relationships. Possibly all those concepts will be put aside. Maybe to pick up at one point later, but not so much part of the Circle notation. I do think that in the Circle projects in the past, I had figured out that the Creator concepts are probably integrable with Circle, though the precise notation was not worked out to the last detail. I remember now, that I was quite keen to try and make sure the Circle and Creator ideas were to be combined into one. Later on I think I had it clearer in my head where these two ideas met, and how they could go together.

Maybe describing it more 'openly' would just make it usable as a proposed construct notation. Right now the aim is not so much that everything will work as a run-time or that automatic determination of containment structure is also let go of, which might simplify things. I just might not want to propose that this is a required thing.

I think I remember that some other chapters do tap into it. Maybe it is nicer to just keep it separated into this topic on its own, so not to make other notation explanations dependent on it.

### 2020-05-18 Postponed Content Changes for System Commands

* Content: System Command Call Notations: I actually feel uneasy about the notations introduced there at all. I want to try and avoid exaggeration, but this seems like a opening a flood gate to ambiguity. But that is not the question at hand.
* Content: System Objects seem to lead to Assignment, Connectors and Connections, but one is sort of the internals, while the other is just basics of notation, so not sure to keep them intermixed like that.
* Content: I wonder if System Commands and System Interfaces could become 1 article, because is the System Interfaces article(s) the visualization of the System Commands? I tend to not like conceptual explanations without accompanying diagrams anymore, so perhaps. I can consider later.

### 2020-05-18 Postponed Content Changes for Commands

* It seems to me that I wanted to evaluate back then:
* What concepts from other programming languages could look like in the diagram notation.
* What each diagram notation situation could mean in terms of other programming languages like C#.
* What would happen if you try to draw a parallel between objects and commands. Where things overlap, what the implications of being a command is, compared to an object.
* That.
* But it may have resulted in a bit of an overwhelming set of situations being shed light on.
* Not sure to what extent this is a problem.
* It's more of a content issue, rather than a formatting issue. I was focusing on formatting issues for now, so maybe postpone this.
* What if you would just remove most of the rules imposed onto commands compared to regular objects? It seems that maybe no disasters would happen. I would like to evaluate which rules can be lived without so the idea might be presented in a simpler way.

### 2020-05-20 Postponed Content Changes for Parameters

* Maybe make the idea of ubiquitous bidirectional relationships between commands and objects less centric. Or perhaps at least diverge into other approaches, so attempting to make room for expressing what goes on in other languages, rather than replacing the construct by another take on it.
* I think maybe that bidirectionality is not of much use in this Circle Language Spec. It seems better off being placed in Circle Construct Drafts, because it just does not resonate much with other languages, programming seems to be able to live without it, and it seems alienating and full of implications, that seems to make concepts fall apart that did not seem to need to be ripped apart in the first place. I might be too harsh about it here. It is an interesting idea, I think, but it just complicates the idea of the diagram notation, which seems centric, not drastically changing how constructs work.
* The Imported Parameter Concepts could be tidied up. The ideas from the original Symbol Language.doc might be processed: discarded or embedded into the main text. Merging the texts of the conceptual explanation and diagram notation might make things more overviewable and shorter / less text.

### 2020-05-20 Postponed Content Changes for Execution Control

* Execution Control sort of seems an application of Circle Language rather than an intrinsic part. Arithmetic and other operators were supposed to be something like that too: an application of Circle diagrams. I wonder why I explain execution control in detail, while for math operations I leave that out. Maybe an article about that would make things clearer/complete/make you not wonder where all the operators have gone in this language.

### 2020-05-20 Postponed Content Changes for Black Boxes

There are topics, I probably want to move away from the Circle Language Spec at one point and go for something simpler. The whole friend notation is something I scratch my head about. I am not sure, if I like it or not.

I am not sure if the topics I dislike can be left in separate articles or something.

* Obs: One of the first few headings is that 'friends' idea in the 'Black Box Main Concepts' article.
* I think it might be hard to keep that separated from the rest.
* Maybe I should forget about trying and go for just longer articles, less articles.
* 'special reference:
  + When I think of a reference with a wavy line instead of a straight line, and all I think is 'special reference' then it clicks. I think I can just read the wavy lines then without much problem.
* I am still quite convinced I might want to introduce some general notation for more conventional member access control situations, perhaps derived from the 'full blown' system and with certain rules just implied, like 'a class's method has access to the private members of the class'.
* Half-way this one I start finding it difficult: Public & Friend Connections.
* From there on there are a few parts that I cannot easily read. I am not sure why. I just do not understand some of it, or the notation for some reason starts to become boggling to me, while earlier it was clear. I wonder whether it has anything to do with the way I explained it, or the idea itself is complicated or something. Maybe something in between.
* I do not worry as much though. I think it is appropriate to introduce a simpler scheme for the most common cases and do that at the beginning. Perhaps saying next, that it is based on a more elaborate notation covered after that, which explicitly denotes things that are just given in certain other languages, or a possibly broader range of access controlling possibilities than offered in other languages.
* I also feel that there could be more than one way to denote public and private with or without connectors. Not showing connectors at all might denote that something is private. Or not showing the members at all might imply things are private. Or showing a connector with a cross might say 'private', but that would imply the ones without a connector would be public? Is it bothersome to show public access connectors for everything public while all you are showing is the public members? Not sure. Maybe not.
* Maybe this detail is not even that important if you just present the options and leave the rest up to the imagination. My goal might not be to use this literally as described. I would be glad if anyone would even use bits and pieces, if at all.
* In the document Circle Language Spec Strategy one of the 'guidelines' for 'writing efficiently' was to not try to compare it to other systems too much. I am beginning to think that it might be useful to do the opposite when it comes to the Black Box principles. I think a demonstration of common cases in other programming languages might be more useful to start with, and then maybe later go into a systematic approach of maybe a broader view on what access modifiers actually do. That last bit might even be more something for Circle Construct Drafts, but I am beginning to think, that access modifiers from other programming languages, could be specific cases, of an underlying more general pattern and perhaps richer set of options. Why I now push that in the foreground in my story, is because I possibly would like to base notations of specific cases on this more general underlying pattern. The friend access thing… I might want to demote that to a special case, or specialized idea of special situations that might occur in other languages (friend access modifier in C++) for instance, and the philosophical idea of consequences of drawing a complete parallel between commands and objects. Maybe that idea might even be too alienating for Circle Language Spec. Maybe Circle Construct Drafts might be where you want to move that over to. But maybe if you just save it for last, and introduce it as the role I think it could have (not necessarily a big one), it might be OK to leave it in there.
* It's just that the Black Box story might be the chapter I am least satisfied with. Not sure yet.
* Note that the Events chapter has an item 'Black Boxed Events' in it.

### 2020-05-22 Postponed Content Changes for Interfaces

* Interfaces Main Concept, section Explicit Interfaces:



has something missing. In C# you can also refer to the parent object as if it were of the type of the interface, right? And it is also not an accessible sub-object, right?

### 2020-05-22 Postponed Content Changes for Inheritance

* Detailed.
* Also raises questions here and there.
* For instance the assumptions question "Where did the language lose its purity?" comes to mind in. I do see that in this chapter's text I am honest about some of these issues. It seems that later texts seem more humanized that way, what I am sort of aim to do in this project.

### 2020-05-22 Postponed Content Changes for Type Control

* It seems part of it is from the old Symbol Language.doc. It seems to attempt to demonstrate how the terms generic and explicit apply to the diagram notation. It seems to attempt to demonstrate any possible variation of it. But in doing so, it seems to explain things already explained elsewhere, but now insisting the terms generic and explicit are so centric they must be systematically demonstrated. That was a bit sarcastic, sorry, but I question if feeling the terms generic and explicit are important are reason enough to explain things twice from a different angle.
* What I sort of expected to see when I opened this document, was an explanation on how type safety might be enforced in Circle. But what I saw was the different forms of generic and explicit, that explain things I already saw in other chapters.
* I think maybe I try to prove in this text too much that the idea of generic and explicit works in Circle.
* I cannot say it is bad. I am not sure where to go with this.
* It almost seems to me, that pieces of these explanations could be spread among chapters Objects, Classes, Interfaces, Commands etc. Type systems seem sort of cross-cutting and centric in object oriented languages.
* I once seemed to feel it was a main issue. Now I am not so sure.
* I do kind of like it. I like systematic overviews. It probably is redundant information. Perhaps I can say that in an intro paragraph eventually. But for overview: I think it's nice. I think maybe I was prejudiced about it being mostly from the previous documentation Symbol Language.doc. Prejudiced about it being old. The argument that it seems redundant info I stand by, but I think it can serves a purpose.

### 2020-05-24 Postponed Content Changes for Object Order

I might move that Object Order issue to the chapter Objects. It's single lone topic now. It starts with the word 'Object', so maybe therefore Objects is the chapter for it. It's just right now (2020-05-24) it is unfinished and unfinished things tend to be put at the bottom of the documentation right now and do not want to do anything about it right now, I think.

### 2020-04-18 Postponed Reword The Done Circle Language Spec Planning Docs

* 2008-05 01 Diagram Expression Specs Project Summary.docx
* 2008-05 02 Classes & Relations Specs Project Summary.docx
* 2008-06 01 Execution Control Spec Project Summary.docx
* 2008-06 02 Command as a Concept Spec Project Summary.docx
* 2008-06 03 Clarify Command as a Concept Spec Project Summary.docx
* 2008-07 01 System Objects Spec Project Summary.docx
* 2008-07 02 Assignment Spec Project Summary.docx
* 2008-08 00 Commands Spec Project Summary.docx
* 2008-09 01 Organize Circle Language Ideas Project Summary.docx
* 2008-09 02 Parameters Spec Plan.docx
  + "Legacy" 🡪 "Imported"
  + Is it still quite harsh here and there? Could be softened up even more?
  + WAS AT: Changed Approach
  + I am reconsidering in how much detail I would want to go with this.
  + I lost track of why I want to do this if it's that much work.
* 2008-10 01 Scatter Symbol Language Documentation Content Project Summmary.docx
* 2009-06 01 Circle Language Spec Plan Part B, Sub-Projects.docx
* 2009-06 02 Circle Language Spec Plan Part B, Evaluation.docx
* 2009-06 01 Planning + Black Box Spec Part A, Eventual Plan, Goal Description.docx
* 2009-06 02 Planning + Black Box Spec Part A, Eventual Plan, Elements.docx
* 2009-06 03 Planning + Black Box Spec Part A, Evaluation.docx
* 2009-06 04 Planning + Black Box Spec Part A, Initial Plan, Goal Description.docx
* 2009-06 05 Planning + Black Box Spec Part A, Initial Plan, Elements & Time Estimation.docx
* 2009-06 01 Planning + Black Box Spec Part A, Notes, Startup.docx
* 2009-06 02 Planning + Black Box Spec Part A, Notes, Cycle 1, Process Idea Box.docx
* 2009-06 03 Planning + Black Box Spec Part A, Notes, Cycle 2, Black Box Part A.docx
* 2009-06 04 Planning + Black Box Spec Part A, Notes, Cycle 3, Redo System Objects & Assignment Part A.docx
* 2009-06 05 Planning + Black Box Spec Part A, Notes, Wrap Up.docx
* 2009-08 Review by Ramses, Small Plan.docx
* 2009-09 01 Wrap Up Black Box, System Objects & Assignment Specs, Goal.docx
* 2009-09 02 Wrap Up Black Box, System Objects & Assignment Specs, Startup.docx
* 2009-09 03 Wrap Up Black Box, System Objects & Assignment Specs, Implementation.docx
* 2009-09 04 Wrap Up Black Box, System Objects & Assignment Specs, Wrap-Up.docx
* 2010-02 01 Interfaces Spec, Strategy.docx
* 2010-02 02 Interfaces Spec, Product List.docx
* 2010-02 03 Interfaces Spec, Steps.docx
* 2010-02 04 Interfaces Spec, Evaluation.docx
* 2010-05 01 Events Spec, Strategy.docx
* 2010-05 02 Events Spec, Product List.docx
* 2010-05 03 Events Spec, Steps.docx
* 2010-05 04 Events Spec, Evaluation.docx
* 2010-05 01 Inheritance Spec, Strategy & Steps.docx
* 2010-05 02 Inheritance Spec, Products List.docx

### 2020-05-14 Postponed Circle Docs Folders Neater

* Make folder subdivision neater, now that the topics have found a new place. ("Circle Language Spec", "Circle Constructs Drafts", "Circle Broader View" and "Framework".)
* Circle Language Spec has priority over the other main folders.
* The others are postponed for now.

## Done

### 2020-05-24 Done Merge More Articles

* More articles from the first chapters might be merged together.
* Structure changes:
  + x Making back ground black?
  + x Changing main headers?
  + / Removing the footers?
  + / Removing 'see also' references?
  + x Convert to docx.
  + Merging or redistributing articles?
* Chapters
  + Objects
  + Classes
  + Relations
  + System Objects
  + Commands

### 2020-05-27 Done Merging More Commands Articles Together

* Redirection and recursion seem to have some overlap conceptually.
* Maybe 05. Comparison to CPU-Like Calls belongs in 04. Creation Behavior Of Commands.
* 01. Commands Main Concepts > Larger > Keep separate
* 02. Start & Stop > Larger > Keep separate
* 03. Parent Controls Its Sub-Executions > Medium sized > Misc-ish
* 04. Creation Behavior Of Commands > Larger
* 05. Comparison to CPU-Like Calls > Medium sized > Might group into Creation Behavior Of Commands
* 06. Command Redirection > Medium sized
* 07. Recursion > Medium sized / smallish > Might group into Command Redirection.
* 08. Command Referrers > Smaller > Misc-ish
* 09. Commands Example Diagrams > Larger > Keep separate
* 10. Commands Ideas > Larger > Keep separate
* 11. This (unfinished) > Smaller > Misc-ish

### 2020-05-26 Done Merging More System Objects Articles Together

* Last time it seems I only merged together a few large groups of articles into their main topic.
* Right now I feel maybe the first 5 could be merged into one.
* Along with some main topics, I feel there are loose topics. Maybe those loose topics could be grouped together with something.
* I think Connectors and Connections is something I might want to combine into one article.
* Assignment I would like to keep separate for now.
* System Command Call Notations too. It's kind of large and honestly a bit questionable, so I would like to keep that separate.
* I am thinking the remaining 4 relatively small articles, might be combined into a System Objects Misc article.
* 01. System Objects > Larger > Keep separate
* 02. Assignment > Medium sized > Keep separate
* 03. System Command Call Notations > Larger > Keep separate
* 04. Connectors > Smaller > Combined into Connectors & Connections.
* 05. Connections > Medium sized > Combined into Connectors & Connections.
* 06. System Command Calls by User > Small > Put into single article System Objects Misc Issues.
* 07. Objects Floating Around > Small > Put into single article System Objects Misc Issues.
* 08. System Command Extension > Small > Put into single article System Objects Misc Issues.
* 09. Parameters For Objects > Medium sized > Put into single article System Objects Misc Issues.
* 10. List Concept > Medium sized > Keep separate
* 11. System Objects Ideas > Larger > Keep separate

### 2020-05-25 Done Merging More Relations Articles Together

I feel merging the articles together is not something I necessarily want to do right now. I am not sure why those two things are connected to each other in my view.

I guess I feel I might want to polish up the way I explain things, before I put effort into merging the articles together at all? Are the articles not particularly small, like some of the articles I merged together before?

Some articles are smallish or sort of medium sized. I think I could just make it one article. That may make it more manageable the way I would like. I do not see my not entirely agreeing with the concept anymore as standing in the way of that.

Still want to change the fonts.

### 2020-05-16 Done Formatting and Article Merging and Distributing

* / Maybe put Type Control out of scope? > Deemed less important to consider right now.
* So, what's next?
* I think nothing to reorganize anymore about the file structure of Circle Language Spec.
* Maybe start with the texts?
* / Option: Convert to docx. > Too many docs.
* Option: If there are still Text Code topics, they might be moved to Circle Construct Drafts.
* Structure changes:
  + Making back ground black?
  + Changing main headers?
  + Removing the footers?
  + Removing 'see also' references?
  + Convert to docx.
  + Merging or redistributing articles?
  + Putting each conceptual and diagram explanation together in an article.
* Chapters:
  + Circle Language Spec Overview
  + Basic Diagram Elements
  + Objects
  + Classes
  + Relations
  + System Objects
  + Commands
  + Parameters
  + Globality
  + Execution Control
  + Black Boxing
  + Interfaces
  + Events
  + Inheritance
  + > The next are 26 documents total. I might do formatting for all of them in one go. None of the folders seem to have large sets of articles anymore.
  + Type Control
  + Object Resolution
  + Conditions
  + Object Order
  + Uncategorized
  + Other Diagram Topics
  + Summaries of Other Languages
  + Previous Versions

### 2020-05-24 Done Merging and Redistributing Remaining Articles

Maybe going into detail about this is not part of the editing phase I am putting these articles through.

This is what the Circle Language Spec Overview calls the 'unfinished' part. So for that reason it seems less straight forward.

It is also harder to subdivide, because it is multiple different issues, unlike previous tasks I did, which were just about 1 main topic.

The question I have for myself is how much is worth doing with regards to these topics.

Some things I'd like to do.

But I might want to be selective about it.

* / Type Safety, Genericity, Explicit
  + Already 1 article. The topic seems to deserve its own spot.
* / 21. Object Resolution
  + Seems all of the articles might be put in one article, because I estimate these are not big issues separately.
  + A lot of the text is preliminary.
  + I do see this as being appropriate as a separate chapter, I think.
  + I do not see much of a reason to put effort into this right now.
  + It's kind of messy and when putting it into one article now, I may lose the overview that the split up into articles gives me now.
* / Conditions
  + These 3 articles could become 1.
  + It seems that if content is still messy, the split up into multiple articles might have value, because it seems to create overview.
  + Just added some comment on other content about Conditions to find elsewhere.
* 23. Object Order
  + Is one article.
  + But not a big one.
  + Makes me wonder if some topics are handy as a separate chapter or better grouped into another main topic.
  + Eventually might move to the Objects chapter, but will keep it in its own folder for now.
* This
  + I think I already had a place in mind for that, to move this article to a different main topic. I think tso Commands, but I think I put that down in the planning doc Circle Language Spec Product List.
  + Circle Language Spec Product List says that it should belong in the Commands chapter.
* Redirection
  + Circle Language Spec Product List does not seem to assign a specific existing Chapter to it. It says 'ref-ness'? Maybe that pointer to pointer situations might all be put in a separate chapter, this might also belong there?
  + It talks about 'target symbols'.
  + There are several target symbol topics in e.g. the chapters Objects, Classes and Interfaces.
  + Maybe that can all go to a chapter about ref-ness, or whatever generic term I might find for it.
  + If it is just about target symbols, maybe the topics are already covered by the previous content.
  + It does seem to be about redirecting e.g. the object aspect and then about tracing to the final target object/class/interface after following the redirections.
  + I do like the idea of a Pointers chapter or something.
  + This content does seem to be covered elsewhere.
  + Perhaps I should just make a folder 'Pointers' and put it there, and a readme just to make a point, that it could just become a separate chapter.
* 30. Misc Diagram Topics
  + I think this is appropriate as a separate chapter.
  + 1. Diagram Expression Introduction > Larger
  + 2. About Programming In General > Larger
  + 3. Diagram Metrics > Larger or will be.
  + 4. Coloring > Smaller
  + 6.0. Example Diagrams Introduction > Small
    - Maybe move to the folder 6.1. Example Diagrams? Maybe call it Readme?
* 7. Summaries of Other Languages
  + Larger
* 32. Previous Versions
  + This folder structure seems off when there is essentially only one version, as one document in it.
  + (There used to be multiple versions in one folder. Those were put out of scope of the Circle Language Spec.)

### 2020-05-23 Done Merging Inheritance Articles Together

Most of them are quite small.

Grouping together main concepts into one article might make sense.

Specialization is mostly small articles, but quite a few of them. But maybe combine those into one article.

The articles that follow do not seem to be many or they are already larger ones I might not want to group together.

* 01. Inheritance Introduction > Smaller
* 02. Inheritance Main Concepts > Smaller
* 03. Class Inheritance > Smaller
* 04. Object Inheritance > Smaller
* 05. List Inheritance > Smaller
* 06. System Inheritance > Medium-sized / smaller
* 07. Interface Inheritance > Smaller
* 07. Specialization > Smaller
* 08. Specialization & Data Replacement > Smaller
* 09. Altering the Member Set > Smaller
* 10. Member Addition > Smaller
* 11. Member Exclusion > Medium-sized / smaller
* 12. Member Inclusion > Smaller
* 14. Detouring Members > Smaller
* 15. Shadowing > Medium-sized / smaller
* 16. Overriding > Medium-sized
* 17. Altering Command Implementations > Smaller
* 18. Command Extension > Smaller
* 19. System Command Extension > Smaller
* 20. System Command Overriding > Smaller
* 21. System Command Shadowing > Smaller
* 22. System Command Extension By Shadowing > Smaller
* 23. System Command Extension By Overriding (Questionable) > Smaller
* 24. System Command Overriding By Extension > Smaller
* 25. Destructive & Non-Destructive Specialization Methods > Smaller
* 26. Misc Inheritance Situations > Medium-sized / smaller > Moved to Inheritance Main Concepts.
* 27. Enforcing & Preventing Specialization > Larger > Would probably keep it separate.
* 28. Deeper Specialization.docx > Larger > Would probably keep it separate.
* 29. Alternate Version Through Inheritance > Smaller
  + > Maybe move this detail elsewhere. Might not be worth an entirely separate article.
* 30.1. Extending System Objects (Older) > Larger
  + > Not sure what the status of this is? It has the word 'Older' in it. Is it deprecated? Covered elsewhere already? I don't know. But at this stage maybe I do not need to know.
* 30.1. Extending System Objects in a Diagram (Older) > Medium-sized / smaller
  + > Would probably merge it with the main article.
* 31. Inheritance Ideas > Medium-sized

### 2020-05-22 Done Merging Events Articles Together

Possibly this is one you want to put into a single document.

Most of them are small.

13. Qualified Event Names should be out of scope, since it is about text code?

The articles before 'System Events' might be put into one article.

I might put System Events in one article.

System Events can almost be demoted to a Construct Draft, but just like System Objects is in there, I might want to keep System Events part of the Circle Language Spec (for now). (The correspondence between System Objects and System Events is that they seem more like part of a run-time or if Circle would function more like a stand-alone language, rather than mostly a diagram notation).

The last 3 articles might become a single 'misc' article. But perhaps some of them have a place in the main article.

* 01. Events Introduction > Smaller
* 02. Prime Event Example, Button Clicked > Smaller
* 03. Events Main Concepts > Smallish
* 04. Event Situations > Smallish
* 05. Explicit Implementation of Event Object > Smaller
* 06. Explicit Interface of Event Object > Smaller
* 07. Event Interface Reference > Smaller
* 08. System Events > Medium sized / smaller
* 09. System Events in Normal Notation > Smaller
* 10. Simplified System Event Notation > Medium sized
* 11. System Event Parameters > Medium sized / smaller
* 12. System Event Design Choices > Smaller
* 13. Qualified Event Names > Smaller <<out of scope>>
* 14. Interaction Events > Smaller
* 15. Black Boxing Events > Medium sized / smaller
* 16. Event Alternatives> Smaller
* 99. Ideas > Smallish

### 2020-05-22 Done Merging Interfaces Articles Together

* 01. Interfaces Introduction > Small
* 02. Interfaces Main Concepts.docx > Medium sized
* 03. Interface Aspect.docx > Small
* 04. System Commands for the Interface Aspect.docx > Medium sized / small
* 05. System Commands for the Interface Merging Aspect.docx > Small
* 06. Interface Aspect in System Interface.docx > Medium sized / small
* 07. Interface Assignment Types.docx > Medium sized
* 08. Interface System Command Calls.docx > Medium size
* 09. Interface Connectors.docx > Small
* 10. Interface Connections.docx > Small
* 11. New Command with Interface Parameter.docx > Small
* 12. Command Interfaces.docx > Small <<use case>>
* 13. Interface Referencing and Redirectioning.docx > Small <<notation demonstration>>
* 14. Member Grouping.docx > Medium sized / small <<edge-case>>
* 15. Relations Through Interfaces.docx > Medium sized <<notation demonstration>>
* 16. Interface Referrers.docx > Small <<edge-case>>
* 17. Reliability of Interfaces.docx > Medium sized / small <<edge-case>>
* 18. Group By Source.docx > Medium sized / small <<out of scope?>>
* 19. Imaginary Backward Relations.docx > Small <<out of scope?>>
* 20. Target Interfaces.docx > Medium sized, because of ideas, main text is small. <<main concept>>
* 21. Mutual Commands, Mutual Interfaces (Unfinished).docx > Small <<construct draft>>
* 22. Extensive Classification with Interfaces (Unfinished).docx > Small <<construct draft>>
* 99. Interfaces Ideas.docx > Small > Move to main article.

None of the articles are that big. The size differences are a bit all over the place.

Logically I might group together all Interface Aspects articles.

The Intro and Main Concepts seem logical to group.

I think I'll just start with those and take it from there.

There's something about text code in Interface Assignment Types. That could be moved to Circle Construct Drafts.

Maybe move the Loose Ideas all to a separate article at the end.

I discuss problems with interface reliability in case of automatically established bidirectional relationships. The idea of always having bidirectional relationships between things is something I want to distance myself from in the Circle Language Spec, and perhaps leave as a topic for Circle Construct Drafts. But right now the idea seems embedded into the Circle Language Spec, and without significant modification, it does not seem you can deprecate it yet. I think such redesign of Relationships may take place in the future (to leave out automatic bidirectional relationships). But I think it is only then that I also adapt the text about this edge cases. So the question that worried me is whether it is out of scope or not. The answer for now seems to be: no, it is in scope, but the design of the concept of Relationships may be changed in later efforts.

### 2020-05-20 Done Merging Black Boxes Articles Together

There are quite a few relatively small articles.

But I have doubts how to group them to bigger articles.

I do not think I want to group them together just in one big article.

I might just go for it, grouping the first smaller articles into a bigger one.

* Smaller, integrated into a single article:
  + 01. Black Box Introduction
  + 02. Black Box Main Concepts
  + 03. Public & Private
  + 04. Friend Declaration
  + 05. Notations of Private
  + 06. Friend Declaration in Instances
  + 07. Public & Friend Connections
  + 08. Access Controlling Aspects
  + 09. Object-Bound, Reference-Bound & Access Control
  + 13. Assignment between Friend Members
  + 14. Friend = Wavy Line
* Group these together:
  + 10. Public & Private Connectors > Kinda medium sized
  + 11. Public & Private Connections > Relatively bigger.
  + 12. Public & Private Assignment > Kinda medium sized.
* Putting these together in one article:
  + 20. Black Box Miscellaneous Issues
    - Looks unfinished.
    - Objects Take Over Class Access Control: Might move to article Black Boxes, section Friend Declarations in Instances.
  + 21. Black Box Side-Issues
    - Looks unfinished.
    - Might move these Circle Broader View:
      * Black Boxing and User Access Control
      * Programmers and Users
  + 22. Black Box Details, May Cover Last
    - Unfinished drafts.
    - Might move these Circle Construct Drafts:
      * Level Limitation
      * Globality Level Limitation
      * Downsides to Black Boxes
      * Get For Access and Get For Copy
      * Inaccessible System Aspects
      * Compared to Traditional Black Boxing (goes in too deep on automatic containment)
      * Other Details
    - Special case:
      * Simplified Access Connectors: I think those ideas could be moved to Simplified Access Control Notation, because that is what they are about. Big brainstorm though.
  + 23. Black Box Details, Might Not Cover
    - Same as above.
    - Moving it to Circle Construct Drafts after some considerations.
* 23.3. Black Box Details, May Not Cover, Simplified Access Control Expression in a Diagram
  + Left it for what it is. Am not calling it unfinished. It seems finished. Maybe might change in the future, but it looks like a worked out idea.
* Moved from Circle Language Spec to Circle Construct Drafts:
  + 23.1. Black Box Details, May Not Cover, Simplified Access Control Expression
  + 23.2. Black Box Details, May Not Cover, Simplified Access Control Expression in Text Code

### 2020-05-20 Done Merging Execution Control Articles Together

* I think I want to merge it into the following set of articles:
  + 1.0.1. Introduction to Execution Control
  + 2.0.1. Conditional Execution
  + 3.0.1. Loops
  + 4.0.1. Jumps
  + 5.0.1. Loop-Related Jumps
* Reason: I had almost merged it into one document, but I like how the separate documents seem to nicely categorize the different kinds of execution control. Also: merging it into one document would give me a tendency to use more heading levels and the styling for those is not optimal yet. Not a strong argument maybe, but it does make me think: More heading levels might be 'proof' that a separation in several articles is not a bad thing.
* I just wanted to save some time merging each pair of documents lots of times only to merge them into a larger document again.
* Taking a look at how it looks now.

### 2020-05-20 Done Merging Parameters Articles Together

* / Maybe merge all the Parameters content into one article if it is not too much.
* Relatively small:
  + 02.0. Parameter Passing.docx:
  + 03.0. Joint Display of Access Connectors and Object Relations.docx
  + Moved their text to the Miscellaneous Parameter Topics article.
* 05.0. Imported Parameter Concepts
  + Not large topics. Maybe Required & Optional is a bit larger. Half of it or so is idea texts, but not bad ones I think. Basically just the version from the previous version of the documentation (Symbol Language.doc).
  + Merging together seems appropriate.
* 10.0. Miscellaneous Parameter Topics
  + Also not large articles.
  + Merging together seems appropriate.
* / I think I want to merge things together in phases, before I decide whether or not I want to turn it into just one bigger article.

### 2020-05-18 Done Merging Commands Articles Together

* I still mean to rename the docs so the 'in a Diagram suffixes are gone.
* Could not see a pattern in what I would want browsing the first few documents.
* Maybe I want to group the first ones into a Main Concepts article.
* I might want to inspect it some more in search of ideas.
* Main Concepts:
  + I suspect I would like to put the 10 documents Commands.docx until Active Clause.docx in one document.
* Hard to group:
  + Start & Stop.docx until 13.1. Changing Inactive to Executable.docx the articles are sort of medium sized and hard to view as a group together, I think.
* Personal feeling: I think the information rate is too high for me now. Lots of topics rush by my eyes at a large pace and my energy level is dropping.
* Even if I do not know how I could group together many of the articles logically, if I can find a few groups, that would already benefit the overviewability of the topic list.
* / These are small:
  + / 15.0. Command References Inside Commands.docx
  + / 16.2. Executables & Executions.docx
  + / 17.1. Procedure.docx
  + / 18. Resolution When Not Allowed For Commands.docx
  + / 22.0. Sub-Commands Not Manually Started.docx
* 19. Parent Controls Its Sub-Executions.docx:
  + Seems like the 3 articles after it are part of it, but not sure.
* Creation behavior:
  + There seem 8 'Creation Behavior' articles. They might not be small, but grouping them to one article might tidy up. Also, it does not seem to me a topic you always want to see in detail. Maybe look at it once and say "Ok, so that's how that could work." Now the topic is quite prominently present in the article list.
* These seem groupable:
  + 35.2. Inactive Command Object Redirection.docx
  + 36.1. Inactive Command Class Redirection.docx
  + 37.2. Executable Command Class Redirection.docx
  + 38.2. Executable Command Object Redirection.docx
* These seem groupable:
  + 39.1. Target Command Object.docx
  + 40.1. Target Command Definition.docx
  + Or maybe put in Command Redirection.
  + Those two have loose ideas in them.
  + I might not want to stop myself from merging articles that have loose ideas. (My argument against it seems to be: It might get messy.)
  + I also might not want to step away from the 'put loose ideas where they belong'.
  + So maybe I want to accept that some loose ideas get spread a little over the content here and there.
  + The idea with the loose ideas was to go through them at some point seeing if any of it is still usable or not and then maybe discard them, or integrate the ideas into the content. Something like that.
* Out of scope? 42.0. Implementation of System Commands.docx
* Example diagrams:
  + Maybe put them all in one article.
  + / Maybe rename and see what they mean.
* Some topics' titles look like main concepts to me. I want to evaluate later if I want to move them to Commands Main Concepts.docx.
  + Maybe if ut were'd for the Start and Stop article in between I would just move more to Commands Main Concepts.
  + Consider merging the content of these with into the Commands Main Concepts.docx:
  + 13. Reading & Writing Parameters.docx
* Merge these two:
  + 17. Command Object Referrers.docx
  + 18. Command Definition Referrers.docx

### 2020-05-18 Done Merging System Objects Articles Together

* Deciding what to do with these, was a bit intense. Especially when I hit a part of the content I have reservations about.
* System Commands:
  + Articles for the various aspects might be merged to one article. Those are like 15 articles or so. Most of them are small. 3 of them are a bit larger.
* System Interfaces:
  + Articles for the various system objects might be put together into one. Not that all of them are small. Just that enough of them are. It seems a bit of a hinder to have to open them one by one. A single article might make browsing the content easier. The amount of articles might be more overwhelming that an article that has some pages.
* Assignment + subtopics
  + May be turned onto 1 article.
* I think the main parts may become more obvious from the article list if I merge together more articles into one.
* System Command Call Notations:
  + 4 smaller, 3 larger
  + Not sure if for the size of them I would merge them together.
  + It is kind of one topic.
* / Misc topics at the end:
  + / Not sure yet if merging is appropriate.

### 2020-05-16 Done Reorganize Circle Language Spec Files

* The bulk of the content is in the Coding Concepts.
* Diagram Expression content:
  + Is now in more than 1 place.
  + "1. Diagram Expression" folder:
    - Describes Basic Diagram Elements.
    - The argument, that the diagram elements must be quickly introduced at the beginning, I am not sure about that anymore. So not sure if this actually must be at the beginning of the documentation anymore.
  + "6. Expression\1. Diagrams"
    - Seems less finished up. I am not sure I would want to put that at the top of the documentation or at least not in this state.
  + "7. Other Topics\9. Old or Boring\About Programming In General.doc"
    - This is old text, that when rewritten I might even want as the introduction text that the Circle Language Spec starts with, or maybe just part of it.
    - Is not really a diagram expression topic specifically.
* / I might want to somehow merge those diagram topics together.
* And put the Coding Concepts sub-folders just in the root.
* In what order will I 'mix' the Coding Concepts and the Diagram Expression topics?
  + Brainstorm: I don't think I want to just start with the Objects chapter. I do like that something starts with the overview. Right now that is the Basic Diagram Elements. Later it may be something else. I think I will keep Basic Diagram Element at the top. Then just the code concepts. Then Other Diagram Expression Topics, just a new folder, a flattened out version of the old.
  + I guess I am just making a flatter list of things and keeping most of the order.
  + 'About Programming In General.doc'
    - Could be put at the top as the introduction.
    - Actually, just reading it over, I think it may be still to rough of a text, that I want to make a bit more neutral, before making it the main thing to start with.
    - It and other bits of texts from the original Symbol Language.doc may compose a new introduction to the Circle language after some reformulating.
    - All in all, I think this content is a but rough for putting at the beginning of the documentation.
* There is a clash in index pages. "Code Concepts.doc" and "Language.doc" may need to become one and cleaned up.
* Moving "Language.doc" content to "Code Concepts.doc" in a controlled fashion.
  + There is no heading 'Object Order'.
* I think I am just going to knead the content.
* The folder subdivision is now much simpler.
* "8.Previous Versions" folder:
  + Has the version of the documentation: Symbol Language.doc.
  + Also contains those Symbol Pictures. I do not want to publish all of them.
  + I do not think I want this document in the published material: D:\Source\JJs Software\Software System Docs\1.1. Circle Language Spec\8. Previous Versions\2004-00-00 XX Symbol Language\Other Files\Coding.txt.
  + Or rather, this has some content in it, that I find questionable.
  + I see two zips with pictures.
  + How do I handle this? In the future I want to go list out all files in the history and remove things from the history that I do not want to publish. Those zips might become one of those files.
  + But when I clean up the current state, will I find all the files back that I want to remove from history? Some have obvious file names. So I guess I will.
  + I question whether I even want to include those example diagrams. Or whether I want to include them in this state.
  + Or maybe I will need them for a better quality referenced image? Nah, I doubt it. That seems too much work, to replace the images in the docs with possibly better images from that zip.
  + It might also not be useful to readers, at least not all of it, because it is just bulky. No one might want to go through them I think.
  + Perhaps a selection of it might be useful.
  + The argument 'Too interesting too throw away'. I might not agree with that anymore.
  + I think most of them are also present somewhere in the documentation.
  + Some of them I want to not have in source control history.
  + Some I might want to keep.
  + If those 2 zips with pictures are the same…
  + I just compared file sizes and they are different. So the zips might not be the same.
  + I could unpack and compare.
  + I think if they are comparable or one is preferred over another, I could remove the other.
  + I could strip one of the copies.
  + Leave that one in the new documentation.
  + So do I remove the one from the 'Previous Version' documentation?
  + For versioning like that, I think it would have made sense to keep the one in the 'Previous Version' documentation.
  + But for control over the content, that I publish, in this case I would really like one copy.
  + Result of diff: 2x Thumbs.db. 4x a file that is not in 'Previous Version', but is in 'Other Diagram Topics\Example Diagrams'
  + Then do I want to remove the one in 'Previous Versions'?
  + I am not sure I want to unzip them first and then commit. Maybe I want to clear it out first after I unzip it, before I commit it. That might make cleaning up later easier.
  + I tried to revert my commit, but it leaves behind files in my working folder.
  + Reason for deleting picture: Don't want it in there.
  + Reason for deleting picture: Is already visible in a document.
  + Reason for deleting picture: It has private info seen right through the paper.
  + Some of them I have a hard time saying goodbye to.
  + I am beginning to think I might just want to remove all of them.
  + Or most of them.
  + The scanned ones probably.
  + The ones certainly used in documentation or otherwise discardable.
  + I think I would remove most of them, and only keep a couple.
  + Oh, I am done.

### 2020-05-14 Done Split Up Circle Docs

* Next step might even be starting to split up the Circle Language Spec itself in 'in scope' and 'out of scope'.
* Doing a git push.
* Renaming git repo Circle Docs to Software System Docs.
* Renaming git repo Project Docs to Planning Docs.
* Moving around folders.
* "Previous Versions\2005-07-28 00 Creator 0.4":
  + For my own overview it might be useful.
  + But for a clear presentation of the Circle Language Spec, it seems to contribute much.
  + It might obscure the whole to include this.
  + An argument for including it, might be to prove it has a development history, but I might not value this that much.
  + Perhaps move it to one of the other parts of the Circle docs than the spec? Broader view or something?
* Branching
* "Language\Integration" and " Language\Libraries"
  + Under the Language folder: might I move those to the Framework part? They are not concept frameworks, but they don't need to be.
* "10. Circle Constructs Drafts\6. Expression\6. More Text Codes":
  + Has the sub-folders "1. Parser" and "2. Text Templates". I do not think those topics belong in Circle Construct Drafts. Maybe in Circle Broader View. But maybe those topics do not belong in Circle Language docs at all. More like framework. The index page includes other sub-topics, that used to be in that folder: such as 'other text codes', 'math' and 'assembly'. So some time in the past those topics were all in one place. I am not sure to exclude all of those topics from Circle Language and then maybe move them to the Framework part of the Software System docs, or whether some of it somehow has a place in broader perspective.
  + They are probably there, because in the main subdivision of Software System, they would fall under 'Code' and that might be what made them end up in the proximity of Circle Language Spec. So it indeed probably does not belong under Circle at all, but it does belong under 'Code' in Software System.
* By now I have:
  + Categorized topics from "Language" over "Circle Language Spec", "Circle Constructs Drafts", "Circle Broader View" and "Framework".
* Checking if I like how topics ended up in one of 3 categories.
* I have deleted files in my commit. I only want to see renames.
* Commit
* Some 'Operating System' topics might belong in a Circle Language topic group.
* Some documents cover multiple topics. I would like to split those documents up.
  + I could just make copies.
  + And then strip out content that does not apply.
  + / To what degree is it useful to rephrase those directory indexes to reflect the contents better? > It might be better to not do that right now.

### 2020-05-10 Done Brainstorm Effort

It may not be within my capacity to do some of the things I plan to do. Thinking critically about word usage and whether I am stating things too strictly… I feel this might not correspond with my energy budget. I made a careful conclusion about that before. Publishing might be more important, than if I might come off too strongly in the text. It feels like this effort is pulling me down more than I might be able to handle. It seems to be weighing on me.

Perhaps I can drop the goal of trying to use a different language style. My next goal was scoping. I seem to still be bargaining with the idea, like: "Maybe I can just reformulate when it *really* seems to harsh." I think maybe that won't work. Because I tend to evaluate the harshness, and this evaluation is 'jittery': sometimes it seems to be 'on', sometimes 'off', sometimes too on tightly, sometimes maybe too weakly. Perhaps I am better off dropping the whole goal of rewording it. I like learning to use language with more air in it, for my personal life for communication to others, or to prevent holding on too tightly to an idea. But this hobby project may need some air in it. Paradoxically, by not loosening up the language. I feel that writing new texts, I already seem to adopt a ligher way of conveying ideas. Evaluating existing texts seems harder.

I think I lean towards the decision of dropping the goal of reformulating texts to be more 'open'. I still have to get used to the idea.

Little reformulations here and there is something I do when reading over and editing my own texts. Some wording changes I might want to be tolerant towards. But the anxiousness might be something I want to let go of.

I feel like I recognize a personality in my own texts, that I adopted to hopefully be taken more seriously. I feel I am on a different wavelength now, or that I want to be.

So with that new perspective on the work, I was done and satisfied with the last page of the text I was working on within like a minute. Perhaps now I can focus more on what I actually chose to do: scope the project some more.

### 2020-03-07 Do Not Do: Brainstorm Circle 3 Programming Planning Docs

(Circle 3 projects are software development projects, unlike Circle Docs projects, which are language design projects.)

It appears around 2010 I started off with programming Circle 3 with the intention of a more rigorous planning methodology and higher quality technical documentation. Around the same time I switched employers. The new employer did not value my doing planning or documentation, just coding. In projects at home it seems I adopted that way of working. So planning docs and tech docs were no concern anymore. I focused on coding. I think I also stopped keeping an hour sheet at home. That felt was freeing. It felt too much like work logging the hours I spent at hobby projects. But the real motivation for the shift in way of working, seems to be that I cannot have 2 methodologies at the same time: one at home and another one at the office. That seems intrinsic to how my mind handles things.

So I have these near-perfect planning docs for Circle 3, while the execution of the projects was almost only coding, no planning, no documentation.

I don't think I want to reformulate the goals of Circle 3 programming projects, to exclude software design. It is not about making those planning docs good.

### 2020-03-22 Do Not Do: Circle 3 Programming Planning Docs

* I could change titles of projects inside the doc content too.
  + Also for Circle 3 Programming, though less importantly.
* Could I just go with it, call it 'Circle 3 Programming'? And if I want to mention in the planning docs that to documentation was of little concern, just do it with a more open formulation, like 'documentation was of little concern' or 'very little documentation turned out to be written in these projects. The focus turned out to be on programming the code.'?
* I could put a remark or something in the Circle 3 Programming docs that I did not do any documentation, even though it was the initial plan to do that.
* I was going over some sub-project docs to check if any documentation was written during those projects.
* Other sub-projects I scanned were not clearly any doc issues in them.

### 2020-04-13 Done Brainstorm Reword Circle Language Spec Planning Docs

* Reformulate:
  + I would want to read over those Done projects content before publishing. I should know what I publish exactly and have evaluated it and made some adaptations possibly. Probably nothing in it is a secret, so you don't need to remove it from source control history, but slight changes might be good.
  + I did not read the content of the sub-projects or the idea box.
  + I think, I guess, I would want to go through content top-down for reformulations?
* Tone:
  + Some things especially in evaluations may seem cocky when I call my own successes very, very good. I don't know if I need to change that. I also use I and you interchangedly when I talk about myself. Not sure if I have to change that. The plans sometimes talk in definites. Maybe openness is better language, I mean more wiggle room in the wording. Not sure if that's a problem.
  + I worry what people would think of me. If they'd think I'm arrogant… maybe I should not worry about that.
  + Cockiness/speaking in definites, and scope/out-of-scope are 2 different things. My intermediate goal now was to change the tone, not to change the scope covered by the documents? There are still gray areas of scope. I think I should deal with that later?

### 2020-01-13 Do Not Do: Brainstorm Aspects / Concepts

'Concepts' are almost exactly like 'aspects' from 'aspect oriented programming', except maybe the idea of whether just about everything can be elevated to become an 'aspect' even things that aren't the aspect oriented programming construct. Even hand-written, coded out aspects, such as those System Aspects in the New Computer Language. Cross-cutting concerns that you couldn't isolate out of the system using an aspect, but are still clearly an aspect from a conceptual point of view. Can new programming constructs be found, that can do that, isolate concerns like that? It is hard to express my ideas about it and explain them well. I don't even have it all clearly in my mind myself yet. Also the comparison requires I know all the details about aspect oriented programming, which I don't.

### 2020-04-13 Done Brainstorm Scope

* Scope:
  + Some content in the idea box is also out-of-scope.
  + Strategy: Goal of the Language: It mentions fundamental principles, that may become out-of-scope. Higher Goal does too. Will I separate these things? So say: it’s a (diagrammatic) programming language. I had thoughts about saving the higher goals for later.

The scope used to be split in two: Language Spec and the rest.

Now I lean towards a split up in 3 or 4 actually: Language Spec / Broader Perspective / Construct Proposals / Other.

This changes things. This puts even more out of scope of Language Spec. Earlier on I thought anything that has to do with language spec could be in scope of this project, but part of its postponed parts. Yeah this seems arbitrary terminology, that seems to mean the same, but I am going with it. Earlier anything language speccy would be still in the project definition, but deemed optional. This split up was by feeling it is part of the language or not part of the language. Now I actually have in mind 3 categories of things that kind of fall under that umbrella 'part of the language': Language Spec / Broader Perspective / Construct Proposals.

I feel maybe I am not finished making the first split up and now already starting with the next?

>> I want to check that.

Also the new split up would give the wilder ideas a place, not on the big pile 'other', like internet as a single computer would be 'broader perspective'. Actually… you could interpret it differently. Everything not part of Language Spec will at first be put on the pile 'Other'.

Maybe it is better to put off fine-tuning the scope for now. First make it neat the way it is. I was making such nice progress with that, wasn't I?

### 2020-05-08 Done Brainstorm Priorities

* Rewording the Done projects is getting a bit boring.
* / Maybe wrap up Parameters planning doc.
* Scoping was an alternative thing to focus on.
* / Something else could be actually digging into the Circle Language Spec docs themselves.
* I think scoping is a better choice for now.
* I want to go over the Circle Language Spec Plan documents, moving things out of scope off the top of my head.
* Side-goal: I read nicer docs, giving my head a bit of a break.
* I think I will mark things with red before actually moving content out.
* Still lots of definites in the wording of the main planning docs.
* I do like too see it reworded it to nothing being a strict rule anymore and you can always make up your own mind. It seems this does not harm the integrity of the idea.

### 2020-05-10 Done Scoping: Symbol Language and Software System Planning Docs

* Scope the planning docs even further.
* I went through all the planning docs.
* Marked in red things in the Main Project documents, things I would like to place out of scope.
* These two documents may have content to weed out at one point:
  + "2009-08 Review by Ramses, Small Plan.docx" has out-of-scope things. It's just a little hard to extract. Perhaps postpone that when I reconsider reformulating this content again.
    - Marked some content in red.
    - Take out the person's name.
    - Still want to split the document up in two.
  + "Diagrams, Coding Principles & Coding Concepts Plan (obsolete).docx": I may want to consider moving this out of scope, or at least part of it.
    - Marked some content in red.
    - Still want to split the document up in two.
* The other Circle Language Spec Plan docs seem fine regarding scoping for now.
* Still want to move the things marked red in the documents, to the out-of-scope documents.
* / Or consider the 'Software System' topics to put in scope.
  + / The content seems of not much use for inclusion in Circle Language Spec, not the planning docs nor the produced 'specification' docs.
  + / Could be that this folder is relevant: "2001 - 2002 Symbol"
    - / It is basically 4 empty folders to indicate projects, that took place, on the time line, without having a description. There is probably notes in hour sheets about this period.
  + / Could be that this folder is relevant: "2003 - 2009 Software System Documentation"
    - / "2007-10 Rethink Software System, Project Plan.doc":
      * There it seems to start interweaving with Circle Language Spec topics.
      * The goal does not really talk specifically about programming languages. The notes taken while doing that project, mention several topics about programming languages, but just naming larger parts, not details. Not really much can be found that goes specifically into Circle Language Spec and the diagram notation. I do not think much is of use in the Circle Language Spec Plan documentation. It seems 'Symbol attempts' is the real first effort put into the diagram notation, that is mentioned in these planning docs (perhaps aside from what's in the hour loggings, which have not been considered yet).
      * I do not think I can even formulate just a few sentences 'this and that' has been done regarding Circle Language Spec in this project, while the project was also about larger parts. It's just that inconcrete.
    - /"2009-04 01 Full Revision of Documentation 2009-04 - 2009-06"
      * / "Software System Full Revision of Documentation, Goal.doc"
        + / Most things are actually about Software System as a whole.
        + / Only in general the Circle Language Spec is mentioned, but no content seems to be concretely about Circle Language Spec.
      * / "Full Revision of Doc, Elements & Time Estim.doc"
        + / "Main Modules: (19 articles, 90 sections)": There is the first time concrete work for Circle Language Spec is mentioned, done during the project "Full Revision of Documentation 2009-04 - 2009-06".
        + / At "- Sub-Modules: (25 articles, 218 sections)" it starts going into a bit more detail about it.
        + / Up until now, just some main parts of Circle Language Spec are named. Not sure if it is really that significant information to tell about projects and their execution.
        + / It is recognizable for me, the topic subdivision. I could extract some info from it later. I could decide later what to do.
      * / "Software System Full Revision of Documentation, Reflection.doc"
        + / Half way, this reflection turns into almost the literal content of what the goal is of the Circle Language Spec project.
        + I think I can remove it. It is basically the strategy doc of Circle Language spec. Consider that later.
        + / It is interesting to see that Circle Language Spec is exactly the next project after this.
        + / So this is indeed the interlude to the Circle Language Spec project. I like to see those connections.
      * / "02. Software System Full Revision of Documentation, Implementation, Reach Goal A.doc"
        + / Has some detail about Circle Language Spec, but not much.
      * / "04. Software System Full Revision of Documentation, Implementation, Reach Goal B.doc"
        + / Among other things, lists main parts of Circle Language Spec again, for which contents/direction 'pages' were written.
      * / "06. Software System Full Revision of Documentation, Implementation, Finishing Touch.doc"
        + / Almost random details.
    - / The other parts of "2003 - 2009 Software System Documentation" contained nothing relevant.
  + / Should any of that be included in the Circle Language Spec Plan docs?
    - / "2003 - 2009 Software System Documentation" is sort of the prequel to Circle Language Spec. Haha.
    - / And the "2001 - 2002 Symbol" phase starts with initial ideas about Circle language, and ends with the unfinished Symbol.doc, though not much was written down about the progress (though probably things are in hour loggings, not considered yet.)
  + / Looking at "1. Software System.doc":
    - / If none of this part of the planning docs is usable for inclusion in Circle Language Spec planning docs, it may be worth a peek at the actual document Software System.doc or the 'redirection pages'.
    - / I like it. I do not think I see much content I would use for Circle Language Spec. From earlier ideas I had, I expected to find more. Perhaps I looked it over wrong. Or maybe now much is narrowed in scope, some more content from this document does not fall into scope anymore, making less of it very usable in Circle Language Spec. Just bits and pieces maybe, but I expect to find texts of similar content and quality near the topics of Circle Language Spec already.

### 2020-04-18 Done Brainstorm Writing Style

* / It kind of bothers me that the story starts in the middle. Where are the other sub-project descriptions? Maybe mixed with other big projects, like 'Software System Documentation'. Maybe.
* Brainstorm: I worry a lot if the words come off too strongly. On top of that I am not in a very subtle mood right now. If I was feeling more calm and friendly, I might have better judgement about it. Maybe I cannot do this right now. I can only judge language constructions, not whether it feels right. When my base line mood would be calm and friendly, I can just feel whether the words bump with that or not. That might be easier. Sometimes just using calmer language, makes me calmer, but right now it does not seem to be working.
* Wording:
  + Is it really such a problem, that I talk in definites about steps taken and rules of the systematics as I try to work them out? Maybe in talk like that, still add a 'might'. Imagine what if you would not explain to yourself, but to someone else that has to work it out further. Imagine what it would feel like if someone else read it. Like you're being dictated rules, perhaps.
  + After that I might want to change command-like sentences and rule-like statements, to just add 'might' to it or other wiggle words. I would talk to myself that way, just maybe comes off harsh. Maybe being less strict towards myself is also not a bad idea.
  + Added maybe's.
  + Perhaps after that a reformulation phase. I tend to come up with nicer flowing alternatives to just the word 'maybe'.
* Brainstorm wording:
  + Maybe I go a different direction with this.
  + With my gut I might assess whether the language is too blunt or not.
  + Maybe if it just list things, that actually happened, then it might not be much of a problem.
  + Definite language to create a pseudo-truth for a false sense of control, might be a problem.
  + Things stated as rules, that are not hard rules might be a problem.
  + When setting rules to follow a plan, wiggle words might be helpful.
  + Black and white opinion about something that may or may not be true, might be a problem.
  + In some sentences it seems more obvious. If they sound opinionated and black-and-white.
  + Maybe I am looking for clues in texts where there are none. Sort of like when I am driving and see no other traffic, I am scared that there may be traffic, that I am not seeing.
  + Words like 'annoying' or 'constantly' trigger more easily. Stating something as definite, without 'maybe's seems a more subtle 'problem'.

### 2020-04-18 Done Reword 2008-07 02 Assignment Spec Project Summary.docx

* Adding maybe's and perhapses.
* Second phase reformulating > Did not change much, but did change a few things.
* The word 'you' seems overly used. Try to reformulate to object-centric grammar, instead of person-centric.
* Maybe read one more time for possible small typing errors.

### 2020-04-18 Done Reword 2008-05 02 Classes & Relations Specs Project Summary.docx

* / Rename "Relation" => "Relationship" > I would keep the term relation in there for now, otherwise I would have to go through so much documentation. If I refer to a product with the work relation in it and change it in the planning docs, then I also would have to change it in the language spec docs and that is just a whole lot of work. I briefly looked up a discussion online about these words, and it does not seem people can find a really clear distinction or rule, just wishy washy hand wavy stuff, I think. It's just that in IT the term is usually 'relationship'. In the spirit of not alienating readers you might change it, but it does not seem semantically incorrect would you accidentally use the word 'relation' instead.

### 2020-04-18 Done Reword 2008-06 02 Command as a Concept Spec Project Summary.docx

* It mentions topics very much out of scope.
* I think it might not be harmful to keep that mentioning in there.
* Maybe I am trying to hide the out-of-scope topics too rigorously.
* There seem to be maybe's and perhapses missing, but it does not seem to sound too blunt.
* Reconsider to split into in-scope and out-of-scope parts.
* Brainstorm: The input / output topics is so present in this sub-project description. It does not seem lightly touching the topic. Sometimes it is half a paragraph that goes on about just that, while it is not in-scope anymore. Maybe it is worth trying to extract it out, so the summaries of the work become simpler.

### 2020-04-18 Done Reword 2008-06 03 Clarify Command as a Concept Spec Project Summary.docx

* I would like to look at the end-result of the project 'Clarify Command as a Concept Spec' to see if the description of the work done still makes sense if you split it in two.
* Maybe the sub-project Clarify Command as a Concept is mostly out-of-scope.
* That whole project should be out of scope, because the end-result is just about input/output and concurrency resolution.
* That would change the super project description too.

### 2020-04-18 Done Reword 2008-07 01 System Objects Spec Project Summary.docx

* Maybe reformulate harsher words, rather than understanding the text of all the notes. The notes are really unpolished.
* > Was at 'Other Issues' taking out the more harsh wordings.
* I see the difficulty of using the words 'concept' vs. 'aspect'. I want to call it 'aspects' everywhere. But contradictory: 'aspects' is now considered out of scope, while I cannot easily prevent mentioning it at all, because it was in the back of my head during some of these projects. I also have this a little with the term 'code base' which I want to rename to 'base of the code', because 'code base' is a term often used differently in IT.
* Out of scope ideas:
  + I see this document talks about what I would rather call a runtime. Figuring out how Circle could work stand-alone up and running is probably not in the scope of this project anymore. Just describing the language, mostly the diagram notation, is.
  + I see a sentence where I call the System Objects essential for the workings of Circle. But if you see the notation as separate from the runtime it does not seem that essential.
  + So how do I reword this putting these ideas out of scope, without harming the integrity of the content?
  + Hyp: In the original text I express that I feel system objects are essential, as a run-time, not for the notation.
  + I keep seeing the idea of a runtime back. It's an interesting idea, but not in the anymore scope as I want to isolate it. I am reluctant to remove the text, because it is so nicely descriptive about that idea.
  + All in all, I am still uneasy about multiple subjects now considered out of scope, being intermixed in the texts of this sub-project description.
  + The 'other issues' seems to contain loose ideas not fully worked out, part of the aspects construct for instance, or about a run-time. It seems to me some of those ideas, e.g. under 'Other Issues' could be cut-pasted to out-of-scope Circle Language projects.
  + It seems the System Objects Spec Project Summary has a some content that is maybe lazily left in there, while they are also like future ideas?
  + Q: The things that I now consider out of scope, that are mentioned in the sub-project planning documentation for System Objects, are those things even inside the actual language spec content for System Objects? Or is the language spec content also littered with now consider out of scope content?
  + The chapter starts as if it *is* a run-time.
  + Maybe you can solve it easily, by introducing it as though it would run as a run-time, but the idea of controlling system aspect is wider in use and can represent constructs from other languages too. Or something. Maybe just put it into context. Or multiple contexts: can be a run-time, can be representation of system commands already part of the language. In a run-time the idea was that an object lives as sort of reflective data, so a reflection-first environment, where reflection data is not accessed through an API, but more like a readily available, language-intrinsic secondary representation of an object. Having full fledged self-reflective object live in a run-time might not be the most optimal performance-wise, though quite flexible perhaps. An idea is to have the system interface be there if you need it, and optimized away if not needed or just be a view on it, not necessarily a non-optimal way of object representation in runtime. I am going too far with this now, I think. Too many edge-cases. Lots of ideas come to mind.
  + I read most of the content of the System Objects chapter. I forgot what for. To assess whether the topics in the planning docs about it even made it into the documentation. Maybe I can assess that later.
  + Read the sub-project documentation again, to see if some things can be moved out-of-scope?
  + As I read the first sentences of the sub-project planning doc, I tend to want to redescribe it, now I can put it into context. As I explain that context, the integrity of the planning doc seems in tact again.
  + I want to keep reading over and reformulating.

### 2020-04-18 Done Reword The Done Circle Language Spec Planning Docs

* Ideas document: ISNSE = Internet as a Single Computer > But why? Internet aS a[N] Single Entity?
* Brainstorm: Scope the Done projects?
  + I was already putting things out of scope, not just reformulating. I have done that for the Future projects, because putting it out of scope was easier than reformulating some of the difficult text. Maybe that rule does not hold up for reformulating the Done projects. I want to realize that the focus is on reformulation, not scoping. Only if it helps me get through the reformulations, I may want to also fine-tune the scoping in the planning docs.
  + So the same argument kind of applies. If I know it is out of scope, I can just move it without further reformulation.
  + Fortunately I am quite clear on what I put out of scope.
  + Q: Will all 'fundamental principles' be put out of scope?
  + Exp: Read some of the actual documentation.
  + A: Yes, 'fundamental principles' is out of scope with the current ideas I have now.
  + By the way: I saw '2. Concurrency' under '3. Operating System' in the Circle Docs. I couldn't find that content before. Found it.
* I tend to want to subdivide all the Done projects into groups, like I did the Future projects. (in the Other folder, not Circle Language Spec).
  + But I get cramps in my face.
  + It's hard, because it means processing many disparate pieces of information, which takes too much energy/strain for me.
  + Feels bad to leave a grouping/categorization half- finished.
  + Can I make this easier for myself? Dim the screen. Dark mode would have been nice, but don't have it on Windows 7. Take it slower. Worry less about (small) mistakes. Accept this will take a while. Forget about everything else?
* Done projects (now out of scope)
  + / 2008-03 02 Fundamental Principles Spec Part A Project Summary.docx
  + / 2008-03 03 Orient in First Four Fundamental Principles Project Summary.docx
  + / 2008-03 04 Computer Language Programmed Within Itself Spec Project Summary.docx
  + / 2008-03 05 Generic, No Generators Spec, Project Summary.docx
  + / 2008-04 02 Get Clearer View over Difficult Topics Project Summary.docx

### 2020-04-18 Done Circle Language Spec Planning Docs: Reword Main Project and Future Sub -Projects

* The aim is to use less definite, more open, language and to take out the personal 'I' form.
* I think a lot of documents do not contain as much prose as Circle Language Spec Strategy, so maybe things are doable after all. Also it does not drain my energy much. It is relaxing work to me. So I am not sure how much it matters, how long it takes.
* / Limitation: I choose to not reorganize/resubdivide the sub projects. I choose to just change the wording.
* Main Project
  + Mostly finished rewriting the content.
* Future Sub-Projects
  + Assignment Spec Update Ideas.md
  + / Automatic Containment Spec Plan.docx
    - Split up topics about Diagram Metrics from Automatic Containment: two topics. Those intertwine in those two documents.
    - Automatic Containment was going to be put out of scope, because it is so difficult and really in the way of explaining how to use these diagrams normally.
    - I said I would focus on reformulating, not on scoping, but I just don't want to reformulate this content, it is so unclear.
    - Would I simply move more to the 'out-of-scope' document? I doubt, because I moved 'Errors' and the 'Concept Construct' from out-of-scope to in-scope but postponed, because they 'felt' part of the language. Now the scoping rules change and I am wondering about how to organize things for a moment.
    - Looking at the folder subdivision of the planning docs, it does seem like it should be moved there. ~~And so does the concept construct. Moved back again.~~
  + Classes Spec Update Ideas.md
  + Commands Spec Update Ideas.md
  + Concept Construct Spec Project Summary.docx
    - Move out-of-scope.
  + Concepts as External Modules Spec Goal (2008-05) (postponed).docx
    - Move out-of-scope.
  + Concurrency Resolution Spec Project Summary.docx
    - Move out-of-scope.
    - 'Advanced Input/Output' is Concurrency Resolution?
    - There is overlap, but I'm not sure. If I read over the product list, I think I'd have to look into it deeper to determine if all of that Input/Output stuff is up for postponement.
    - I have looked at multiple sources of information, but cannot determine yet if the topics mentioned under the Input Output section of the Products List are in scope or out of scope, due to the concurrency resolution's being in-scope and out-of-scope.
    - / I feel part of the TODO products are already done.
    - / I feel 'normal' parameter topic are intermixed with concurrency resolution / automatic execution order too much?
    - Or is it to shed new light on these topics in the area of automatic execution order?
    - / Alternative: Split apart in two: The parts / product list items, that I know are about concurrency resolution / automatic execution order would be put out of scope and then parts I am not sure about, would stay in in-scope with some comment shedding light on that doubt.
    - Exp: Read over a product list in the Future Sub-Project document "Input Output Spec Project Summary.docx".
    - Hyp: All the topics under Input / Output inside the main "Circle Language Spec Product List.docx" are not trivial parameter topics, but more a small ramp of parameter topics leading towards the automatic execution order concept.
    - / Hyp: But the Integration of Parameter Input/Output Concepts do seem already done topics.
    - Hyp: But may fall under the same umbrella: now put into the light of automatic execution order.
    - Parameters and input/output concepts entangle. The proposed product list would aim to disentangle those topics and explain their precise relationship.
    - I had a brainstorm. I put that in the Future Sub-Project doc in the out of scope folder.
  + Diagram Metrics Ideas.md
  + Diagrams, Coding Principles & Coding Concepts Plan (old).docx
    - At first glance, this seems to have the same scope as the whole project Circle Language Spec.
    - Maybe it can be thrown away (or archived), because it's already described by newer documents.
  + Execution Control Spec Update Ideas.md
  + Globality Spec Update Ideas.md
  + Objects Multiplicity Spec Ideas.md
  + Objects Spec Update Idea.md
  + Parameters Spec Update Ideas.md
  + Relations Spec Update Ideas.md
  + Static Spec Idea.md
    - The second half is an idea, that is explanatory on some of my ideas about static. But also it uses a concept I want to put out of scope: Code = Data. I want to save such concepts for 'Circle Language JJ's Construct Proposals' or something.
    - I would not want to remove it, if I look at that it explains how I see 'static' in a broader view.
    - But to understand my own doubts about the different ways to see the concept of 'static' that idea description explains a lot and otherwise I might think: where did that idea go and how does it work?
    - I feel that this concept of static might have a place in the general descriptions, but to apply this to the concept of Code = Data, though interesting to me, might not have to be part of that, because it may confuse and alienate the reader, who might be more familiar with more regular implementations of the concept.
  + System Objects Spec Update Ideas.md
  + Text Code Spec Project Summary.docx
    - Out of scope
    - But change the main project description too.

### 2020-04-18 Done Circle Language Spec Planning Docs: Format the Project Summaries

* Change formatting in bulk (of Done Projects / Project Summaries)
  + Change fonts and layout in bulk first.
    - All normal text Calibri 11.
    - Heading 2:
      * Use heading 2 for e.g. 'Goal' and 'Date & Time'.
      * Heading 2 Calibri 10 bold, 9pt before, 9pt after
      * Instead of 'Dates' say 'Date & Time'
      * Americanish heading case usage
    - Sub-title:
      * Author and location Calibri 10, italic, 0.5 cm indented
      * No 'date' subtitle, since it is implied by the content of heading 1
      * Make spacing uniform.
    - Indent of normal text is not consistent among documents.
    - Heading 1:
      * Heading 1 month in separate line
      * Heading 1 no comma's at the end of the lines
      * Shorter heading 1
      * Heading 1 Calibri
    - Maybe double-check the formatting afterwards by opening all documents on top of each other, closing them one by one, to see if all is uniform.
    - Forgot the Done projects that have their own folder.
      * > They use different templates, so this formatting and making consisntent, probably does not apply to those.
    - Don't Future Sub-Projects also have documents in that 'project summary' format?
      * Alternative: format those too.
      * Alternative: Go look for more project summaries in the Circle Language Spec Plan folders.
      * Alternative: Don't do it.

### 2020-04-15 Done Circle Language Spec Planning Docs

* I finished 'Circle Language Spec Strategy'.
* 'Circle Language Spec Product List' seems finished.
* The word 'Legacy' is in there, which I tried to avoid.
  + Integration was not the best word, it seemed.
  + 'Inherited' is too ambiguous a term.
  + It is because other languages have those concepts?
  + Oh, wait, it is my own subdivision.
  + Maybe the word Legacy is ok there, because there is only the danger of putting down my own work, not others'.
  + It seems like it could be both.
  + 'Existing' is a little ambiguous too.
  + Why was integrate such a bad word according to my brother?
  + Synonyms?
  + I am going for 'integrate'.
* 'Circle Language Spec Steps & Time Planning' should be next.
  + Yeah, has definite language, that might need more wiggle room.
  + Sometimes I don't even notice it, because I am so used to try and gain a sense of control over the situation, that I just speak in definites.
  + Resolute language is solved.
  + 'Overview for Progress Monitoring' is not the same list as the scored and prioritized work items above it.
  + Why is there even a separate list for 'Progress Monitoring'? It seems handy for quick overview, but why is the main list not quickly giving that overview then? Improve the main list instead and remove the 'Overview for Progress Monitoring'? Because it might just not be handy to have two things.
  + Inheritance is already finished.
    - Do I update that now? Or do I see that as replanning that should be done later? Don't know. It just looks inconsistent this way.
* Done Circle Language Spec Ideas.
  + The wording is in rougher format, but it is not very long, so should be doable.
  + It may have out-of-scope content.
  + Maybe read over again. Second time around it still seems kind of resolute.

### 2020-04-13 Done Circle Language Spec Planning Docs

* I made a slight beginning with a next phase of scoping the project. But I want to turn back.
* Check if I can check this in.
* Do reformulations while maintaining the current scoping.
* This is a lot of work. I am not even sure if I can do it.
* It being forbidden to use the word 'I' and to be afraid to come off cocky… those are very strict rules to set for myself. Maybe too strict. I have trouble with this. Time may be better spent on something else.
* It's weird that if you want to make a project public property, it is strange to have the word 'I' in a project definition. In notes, maybe, but in central project definition it may be weird… But changing this just conflicts with my lack of energy.
* In my efforts to speak in a non-personal tense, I sometimes switch to the 'you' form and then it just sounds like I can't even keep the terms 'I' and 'you' apart… maybe I am over-conscientious and insecure here. 'You' is fine if you explain how one might experience something.
* I read some articles over personal and impersonal forms of language like that.

### 2020-04-09 Done Explore (Content) Search Options

* Exp: Visual Studio Find in Files
* Obs~ Some paths are too long…
* The file had a path of 260, which I think is the max for Windows 7, but some API's use 255 as a max.
* Exp: Shorten to 255. Find in Files again
* Obs~ Error is gone.
* Exp: Get rid of an intermediate folder to shorten the path and still keep descriptive names.
* Obs~ Visual Studio Find in Files will not search doc contents.
* Hyp: Windows 7 File Explorer will not find whole words. For some things that's relevant for other things it is not. (I thought I saw it).
* Exp: search "\*.doc"
* Obs: Now it does not match "\*.docx".
* Hyp: I thought I saw that some times.
* Exp: Windows 7 File Explorer. "D:\Source\JJs Software\Project Docs\Circle Language Spec Plan". Search "content:Joost".
* Obs~ It shows 2 files, if I open them (docx's) and search for Joost in Microsoft Word, I get no result. Odd.
* Hyp: File properties?

### 2020-04-01 Done Circle Language Spec Planning Docs

* Circle Language Specs Done projects:
  + It is a lot of content. It is not practical for me to go into detail about it now. It is only the past plans, not the future, so less important.
* Details:
  + Get rid of content about 'studying'?
  + Command as a Concept Spec Plan seems a super project but later turns out not to be.
    - Actually, it is a parent project.
  + Done project "Document Internet as a Single Computer" is out-of-scope…
    - Lots of fundamental principles may become out-of-scope of the Circle Language Specs Project and might become part of a separate piece of documentation Circle Language Broader View or something.
    - Not sure if right now I want to dump it in a Done folder with all these many projects in it.
    - Does it matter? I am trying to isolate just Circle Language Spec from Other / anything else.

### 2020-04-01 Done Circle Language Spec Planning Docs

* Empty project folders:
  + I could evaluate whether those empty project docs folders are needed.
  + To me it seems odd now, that 'Errors' is put out of scope.
  + I think I want to move the topics 'Errors' and 'Concept Construct' from the Out-of-Scope document to the Circle Language Spec Planning docs.
  + The 'Out-of-Scope' document os currently not just postponed items of the Circle Language Spec, but more than that: not even considered part of the Circle Language Spec proect at all anymore.
  + It is still not a hard rule to be set. But putting Errors and Concepts out of view, makes it sort of not the same project for the sake of Concepts and putting my head into the sand when it is about Errors. If I have ideas about these things, they do belong in the project I think. I have a hard time thinking them away.
* Formatting:
  + Phase-by-phase.
    - Background black.
    - Name spelling JJ van Zon.
    - No abbreviations in folder name and file name.
    - Rename 'Document' to 'Spec'?
    - Suffix with 'Plan'.
    - Do the above for the Current and Future Projects
    - 'Project' => 'Plan'
    - Leave out the word 'Plan' if obvious.
    - Headings 'Circle Language Spec' + similar to folder and file name.
    - 'Computer Language' => 'Circle Language'
    - 'Articles' => 'Spec'
    - No sub folders, just files?
      * Some sub-projects contain many files.
      * Flattening it out might make a big list.
      * Keeping some folders but have other as loose files, makes you lose that time-line feeling.
      * Too much doubt -> Postpone.
    - Maybe prefix all with Circle Language Spec Plan. Then simplify the rest of the title, because Plan is already implied.
      * Getting path length problems. Collapsing sub-folders with only one file after all.
    - Not sure if that prefixing looks so great.
      * It looks cluttered. And I cannot apply it consistently, because of path length problems. Branch. Try without prefixes.
      * 'Spec Plan' somewhere in there looked ok.
      * My doubts are that if this doc is in the internet, the title means a lot, so why not have it be complete? It does already have that completeness somewhere in the URL, but…
      * Maybe let go of the toilet role principle and just call something 'Automatic Containment for Relations.doc'. Not 'Plan', not 'Spec' just lazy. Maybe it is ok.
      * My being better at systematics than words might be in the way. So that I don't easily decide what's clear enough.
    - Convert to docx without Compatibility Mode.
      * What use is that, if I want to convert it to MD instead?
      * Conversion to MD is not great if I use color coding it the docs.
    - Remove '& Copyright'
      * > Only found 1 occurrence: 'Circle Language Spec Ideas'.
    - Remove unnecessary title page white space and 'version XXXX-XX' mentioning.
    - Make main heading and such not take up so much space.
    - Month in the headings, so it is clear that the fact that it is a time period is leading.
      * In the front or the back of the sub-project name?
      * In the file names too in case they have a folder?
      * ~~Months in headings: not entirely convinced about the difference of the order where the month ends up between 'Project Plans' and 'Project Summaries'.~~
    - 'TODO' => 'Notes'? > Cannot find it.
    - Set proofing language to English US. > I don't see the problem anymore. Maybe the conversion to docx fixed it.
    - First pages layout
    - Page header and footer
      * I am good with no page headers and footers.
    - File properties (author and such)
      * If they are odd. Not that odd 🡪 Leave it there.
      * Still have to do Done projects.
    - Update lists of contents
    - Legend:
      * Brainstorm: I might keep the legend as it is. It may be more work to check which colors are used where and to change it? Maybe not, maybe that would be the right way to go.
      * 'Legend' => 'Legend of Symbols and Markings'?
      * Use simpler color coding.
      * There is ambiguity between coloring of back then and now.
        + Brown meant I think postponed (indefinitely). Some share of brown even meant 'part done, part postponed'. Now brown means do not do. (And part done, part postponed is orange.)
        + Light grey used to mean do not do. Now it means done.
      * Normalize the legends.
      * Make it 1 central document
      * This document has a legend right inside it: Concepts as External Modules Spec Goal (2008-05) (postponed).docx
    - Font Calibri
      * Just the normal font, not the headings.
    - / Headings do not have colors.
      * Are not going to work for me anyway, with the low brightness on my screen.
      * It starts being not so functional anymore, the changes.
* I somehow don't trust this is all there's to it about Circle Language Spec planning docs. Look over the state of it. It *would* be fun to move on to a next topic and have something finished. But somehow I don't believe it. > It is. I lean towards the actual language spec rather than the planning, now.

### 2020-04-01 Done Circle Language Spec Planning Docs

* Circle Language Spec Products doc: Move postponed work from Done section to the Postponed section.
* The document "Circle 3 Excluded Requirements" has topics that might be interesting for the Circle Language Spec project.
  + Was at: Round Up looking for things to add/move from Circle 3 Programming to Circle Language Spec.
* Circle 3 Programming, Preliminary Designs:
  + Yes, definitely quite some useful texts in that document, that belong in the Circle Language Spec project.
  + The texts seem to belong in the actual Circle Language Spec docs, not its planning docs.
  + In the past I made effort to dump even unfinished ideas right inside the docs.
  + I had doubts about it, since it would litter the docs with unfinished, lower quality material.
  + But I remember doing it, because otherwise loose ideas would be spread over numerous sub-project documentation and would get out of view.
  + Do I move these texts to the appropriate spot in the Circle Docs now?
  + Do I also move ideas from Sub-Projects there? Maybe those are too short to matter. And not really loose texts, but goals. Somewhere in the middle. So maybe I should copy/move some texts to the loose idea boxes of Circle Language Spec.
  + I'm staring at 'Concepts' now. But Concepts is out-of-scope. Very out-of-scope. Does it matter? Any content about it is to be dumped into the appropriate spot in the documentation.
  + Object Order: Some of it is in Data Concepts. But I fundamentally need it in the Circle language. It's not just list order. It's more fundamental. It can be command execution order, that is dependent on it. I just don't know where it belongs in the chapters. Maybe just its own chapter I was making a separation between coding concepts and data concepts once. Now most coding concepts are just part of the programming language and most data concepts are not. Just the separation is not as precisely clear as that.
* Circle 3 Programming, Requirements:
  + Has technical design requirements.
  + I could look at them, take them out, compare them to the Circle Docs requirements, to see if they are in there or have a place in there.
  + Nah.
* Concepts construct:
  + I am missing writing the Concepts construct documentation from the planning docs. I only see it in the Circle 3 Programming, Excluded Requirements.
* Lower contents:
  + The idea of lower contents is described in project docs not in Circle Docs Planning, but in '1. Circle 3 Programming' planning.
  + You can also find some texts about it here: Circle 3, Lower Contents, TODO.doc

### 2020-03-22 Done Project Names

* Inspecting the sub-projects just to find whether documentation was of any concern in those projects, might be too intense for me right now.
* The question I was trying to answer with that was: Is it accurate to call the super-project 'Circle 3 Programming'?
* The reason for calling it that, is to make it clearer what the project entains, separating it better from the super-project with the name 'Circle Docs', so that there is a clear distinction that one is about programming and the other is about documentation.
* But from the top of my head I kind of already know that the focus of those Circle 3 projects was programming, not documentation.
* I just want clarity on the distinction between projects Circle Docs and Circle 3 Programming, but giving it a clearly distinctive name.

### 2020-03-22 Done Organize Planning Docs

* I moved around folders, for a preliminary split up between Circle Docs and Circle 3 Programming projects.
* I could change titles of projects inside the doc content too.
  + I am unsure whether I want to call the projects 'Circle Docs'. Maybe 'Circle Language Specification'.
  + I do want to go for the name 'Circle' as opposed to 'New Computer Language'.
    - I don't mind that this makes my documentation stick to a name that in theory might later change. It probably will not change at all.
  + But Circle Docs is too general, I think. It should be Circle Language Specification.
  + I don't care for the abbreviation 'Spec'. I am not a fan of abbreviation and I don’t think it makes it much more concise or clear.
  + Also: I am hitting my limit here. I'm tired and getting all sorts of symptoms.
  + Circle Language Design is also an option. 'Specification' may raise expectations about finishedness.
  + Circle Language Design Planning
  + Circle Language Specification
  + 'Specification': suggests finishedness, but does it? CSS is all sorts unfinished modules.
  + Leaving out 'Specification' or 'Design': Makes distinction with Circle 3 Programming hard to see.
  + The word 'Circle' should be in it.
  + The word 'Language' at least sheds light a bit on what it actually is.
  + But Circle Language Specification Plan is just a long name.
  + But 'New Computer Language Functional Design' was too, so that is not an argument?
  + 'Spec': Is an abbreviation. I generally dislike those, but I think in this case it is appropriate. It is adopted as a word by the community.
* I could evaluate whether those empty project docs folders are needed.
  + The 'update … articles' sub-projects
    - Defined as empty project doc folders: I see some of it back in the Circle Language Spec's Strategy doc, Project Steps and Time Planning, but not in the Products doc. Some redo's are defined, be it not in Products doc. It seems the reason for redoing is not always stated. I think I should make the list complete in all the planning docs where appropriate. Then I can get rid of the remaining empty sub-project folders.
    - Mentioned them in the Products doc now.
    - Removed the empty project doc folders.
    - Look at Circle Language Spec Strategy for things to put in Circle Language Spec Products.
    - Look at Circle Language Spec Project Steps & Time Planning for things to put in Circle Language Spec Products.
      * There is a lot there that specifies the reasons to change the articles.
    - Consider that some elaborations in the Circle Language Spec Products document might belong elsewhere.
      * Do they belong in Circle Language Spec Strategy?
      * Sort of collides with the Sub-Project folders idea. Maybe details belong in a sub-project folder and a general reference to it in the strategy doc.
      * Maybe go the other way: put sub-projects strategy in the main strategy doc and get rid of separate sub-projects. (My gut says: no, keep sub-project descriptions. That's what they are for: elaboration on a sub-project's scope.)
      * Decision: Rule: Any elaboration now in the Products planning doc, belongs in a sub-project document.
* Sub-Projects:
  + The documents could probably be shorter. Lots of those Project Summaries are not so useful template texts, not filled in. The notes at the bottom would do.
  + Then it is basically just loose ideas, that might have been given a too prominent place. Maybe put it elsewhere, if it is just unhelpful for visual overview, perhaps. > Some are more than that. I keep em this way.
* I like that I seem to build tolerance (back) for not exactly knowing where I am going with it, gradually shaping it, and just taking my time to make things more overviewable, regardless of whether someone might think it is time well spent or not. I myself just create more overview and understanding of the material that way.

### 2020-03-16 Done Circle 3 Requirements Docs

* I moved content from Circle 3 Requirements to New Computer Language Products doc.
* I could rename it to Circle 3 programming, rather than Circle 3 software development, because programming was all I did, not full blown software development cycles. Right? ('Programming' is a bit ambiguous too. It could mean program the dev environment or program using the new language. But I am OK with it.)
* Maybe check sub-project docs later to verify that I didn't do any documentation.

This is spreading my attention over too many different things. Is there a more practical approach?

This is too intense. I have to stop again.

### 2020-03-15 Done

I read over notes to know where I was, and mark some things of as 'done'.

This was too intense. I wonder if starting to change documentation will be too intense too or if I should stop now. Or if other activities or non-activities will be even more intense. I choose to stop for a while now. After a little over 20 minutes.

### 2020-03-08 Done Reading Circle 3 Requirements Docs

The requirement group 'Priority C: Classes' has 1 language design feature: <Diagram Notation Design> Static. So that is to be moved to the language design project docs.

'Priority E: Integration' contains brainstorming instead of a list of items.

(I notice I get inspired to like implementation projects for Circle 3. I feel the enthusiasm in it. I somehow stopped working on it, though. Also it is not my goal right now. The goal is publish Circle language design eventually.)

Some of the content in 'Priority E: Integration' could be part of language design, at least diagram metrics design and automatic containment.

'Priority F: Large Amounts of Items': Spiraling could be part of diagram metrics documentation. And object order.

'Round-Up': Most are documentation issues, which in theory could be moved to the language design project instead. Except, they are 'technical design' documentation issues, which do not have a place in either the language design project or the circle 3 programming project, because 'technical design' I did not consider language design, because I would have called language design 'functional design'. 'Technical design' according to my views back then, would have be document how I implemented things in the programming of Circle 3, not how the language functionally works, but how Circle 3's .NET code works internally. However, I could see the topics up for 'technical design' as topic that also could use an update to the functional design. So I could consider those topics for extending the Circle language spec project's requirements. Then I could consider removing documentation issues from the Circle 3 programming project docs, because I wasn't going to do them anyway.

I ran over all the content of Circle 3 Requirements and above are the conclusions of what to possibly do.

### 2020-03-07 Done No Planning or Docs Back Then

Circle 3 Strategy is pretty much done.

Circle 3 Requirements: change coloring and formatting.

That does not take away I want to split topics in these planning docs between language design one one end and programming at the other.

### 2019-12-29 Done Brainstorm Scoping

* What do I do with things, that are out-of-scope? Do I just bluntly remove them from the documentation, or do I go through the trouble of parking the texts elsewhere?
* Would I rename 'Computer Language' to something else, admitting it is a programming language, and only expressing the hope that it would become a language to a user to, where constructs are simpler. Do I simply admit that these were my ambitions with the project, and if people claim arrogance, then let them?
* I am hoping at some point, the project docs get smaller… because these documents are huge and intimidating.
* Maybe I should just make 2 project folders eventually in the Project Docs repository: one for the new computer language and one for the rest, that are much like eachother, but one stripped down to computer language functional design topics, and the other in which to dump the rest: anything deemed out-of-scope of the entire new computer language topic. Those are different than topics out-of-scope because postponed, but still much to do with the new computer language. Maybe at first, even 'worse', I make 2 documents in each folder: One with topics that belong to the new computer language, and another document much like it, in which the rest is put, that I would want to leave out of it.
* I think a new concept to me, introduced in this new project is that: I do not need to do everything. Like this from "New Computer Language, Products.doc: "You have to be able to introduce new basic data structures and give them the nonagon symbol, and have different kinds of possible indexers, etcetera." I don't have to. I could do without. Ideas might be viable and interesting without all details being covered, without all proofs being given. I wanted to work out *everything* at some point. I also was a afraid, that if I didn't, people would not believe in the idea. Maybe I got over-ambitious, because I saw so much potential. I think I was able to work out a lot, but then I would get distracted by another project and then it turned out, I never got back to it. Scoping is a trick for that, when managing projects. Setting the boundaries and limitations of what the project would cover. I never wanted to do that back then. I wanted a framework in which everything would fit and then choose seemingly randomly what I would cover next. In one way I like the freedom of that. But on the other hand, it becomes a never ending story. I sometimes had the ambition of actually making *all* of it. I might have been able to create a playground in which I can go wild, but someone else would never want to cover all of that. Someone else would never take over your programming life, just a scoped programming project and then maybe. So I want to scope it. And lose the 'programming it out' part. And loose 'it is also a framework and an OS and any commonly used application'. It is actually quite hard for me to let go of that idea. I liked my playground back then. I wanted proof, that this could be used to realize software quicker, so one man can do what would have taken an army of programmers to do before. But I don't have that ambition anymore. Right now I just want to publically give away the programming language idea. I think I notice a lot of insecurities about people thinking it is a good idea or not. Maybe because I was trying to sell the idea, rather than just give it away? I get that I wanted a framework into which all of my ideas fit. I like some of the modularization of the concepts. But I do want to just cut away a few things. I think I am still trying to sell an idea, but then in a different way. I do not have the intention to sell it for cash, but I do want to not make it too ambitious, cover too much, so large in scope, that no one would pick it up anymore.
* The time planning document ("New Computer Language, Project Steps & Time Planning.doc") looks far more overviewable and less intimidating. It all seems so manageable there.
* The document with the list of products ("New Computer Language, Products.doc") is overwhelming, because each article written is mentioned separately and that means almost each paragraph of produced writing is mentioned separately. If I would just mention the basic outlines, this might be better. Earlier, back then, it may have helped me see what I did and see how much I wanted to do. But with the goal I have now, I think it loses its purpose, and simplicity of the planning docs is more important than rigorous tooling for detailed planning of my own work.

### 2020-02-23 Done Reading Circle 3 Strategy

An idea for today would be to read "Circle 3 Strategy". That document is supposed to be about software development, not language design. At least, that is the new goal I have with that document. Some things in it are relevant for the Circle Docs.

"Goal of the Language" is where it is part about the language design, not so much the software development. I might use/move this text to the Circle Docs project docs.

Was at "More Tips" processing things.

At what point am I going to be more rigorous in splitting off the Circle Docs project docs from the rest? Not yet, I think. It's still a mash up of both in the project docs I am reading now.

### 2020-02-20 Done

Today finished splitting time planning and projects step into in-scope and out-of-scope documents.

### 2020-02-16 Done Notes

Working on splitting time planning and projects step into in-scope and out-of-scope documents. (worked on it for 45 minutes this day.

### 2020-02-13 Done Notes

Working on New Computer Language, Products.docx:

I do not want to necessarily want to shorten the list of products any further. Even the list of Done work. Because some parts are part done and it would be easier for me, would I work on it, to have the same subdivision of the pieces TODO as the pieces that are done.

What I do might want to change is the rough order in phases of the stuff TODO.

I might just go over it a few times, reformulating.

I am splitting off parts of New Computer Language, Products.docx into a separate document with the out-of-scope things.

I want to go over the Postponed topics, to see what content can be moved to the 'out-of-scope' document.

Now I still want to weed out the 'Topics Roughly' moving things to the 'out-of-scope' document with the products.

Also the Strategy can be stripped of things 'out-of-scope'.

### 2020-01-13 Done Notes

I read and reformulated some texts from "New Computer Language, Strategy.doc". 2 hours or so. I am now too tired.

That document does not cover many things out of scope of language specification. It just briefly talks about programming experimental versions and licensing it and stuff, but little enough to keep it in there, were I to isolate this into a pure language specification writing project, which I intend it to.

### 2020-01-30 Done Notes

I read over New Computer Language, Strategy.doc and reformulated stuff.

I am cleaning up New Computer Language, Products.doc: simplified color coding, removed mentioning 'in a Diagram' and 'in Text Code' article variations. I might remove detail from done work, but keep it in the proposed work. May remove some 'musts' by 'mays'. Might add intro docs to calm the reader's nerves down, on the overwhelming amount of topics. Do I need to excuse myself for introducing topic names, without actually describing what it entails? Don't know. That description would *be* the product. I have a canundrum. I cannot describe the product without making the product, because the description is the product.

### 2020-01-04 Done Rough Plan

Rough plan:

* Remove detail from products doc.
* Split main project docs into 2: content about the diagrammatic programming language and content outside of that subject area.
* Or read some sub projects docs.

I had those plans with it, but did nothing about them this day.

### 2019-12-15 Done

* I read over the document "New Computer Language, Strategy.doc" in full and did some reformulations, also removing my never realized studying goals.

### 2019-08-10 Done Brainstorm complexity in preserving rename history

The version folders I have left to convert to source control history are complex, if you also try to get a rename mapping in check. Even if you ignore the rename mapping, you have quite some work to do.

It was not expected that there would be such complexity in preserving rename history, but noticing that there is, makes me re-evaluate my plans.

To get overview of the amount of version folders still to cover: The version folders are visible on the 'root' level per chapter. The amount of version control on *sub-*topics is limited to just one (24. Creation Behavior Of Calls). Its about the *main* topics: Commands, Parameters, Globality, Execution Control, Black Boxing, Interfaces, Events and Inheritance. One of those topics has intensely many version folders (Black Boxing). About 3 topics have a 'normal' amount of version folders. The others have a quite small amount of version folders. If you ignore trying to preserve rename history, then you might be done today. If you try to preserve rename history, it will take you probably many days, like 4 or something. And I think the motivation will recede if I do that.

So I have already made my choice. I will not make effort to preserve rename history. I will just methodically convert the version folders to source control commits.

### 2019-08-11 Done Basic Math Conversion to Source Control History?

The 2 versions' contents:

2005-01-06 00 Former Documentation:

Any programming topic about math I had at the time is thrown into one document here:

\* JMath 0.9 docs in Dutch

\* Simple Math operators

\* Ideas about 'regulated systems'

\* Number Bases

\* Brainstorm: Some alternative wordings and loose ideas written down.

The XXX version is only Simple math operators and how they can be implemented as objects.

Idea bout converting version folder to source control history:

So they are not mutually exclusive at all.

I have doubt whether I should even do the conversion from version folders to source control history here.

Using my rules, I'd put the older docs in archive, but neither 'version' is more deprecated than the other. They are both old, and they are both the latest version of things.

The math as objects does have a link to how math can be made not intrinsic to the language, but an extension library, which can still be compiled to good old CPU instructions. That concept is interesting for the idea of the new computer langauge. But... none of this documentation is well worded to suppord that concept...

### 2019-08-11 Done Controls Concepts Conversion to Source Control History

* I can convert the version folders to source control history the regular way, because each successive version does seem to replace the former version.
* Where is that navigation model brainstorm?
  + It is in the control concepts' root folder. It has a doc in it directly, which I overlooked.

### 2019-08-05 Done Brainstorm Restructuring Docs

The Circle project docs took a turn at some point in time. At one point it was mostly about documentation, then it became about both documentation and programming. But the project docs folders do not seem to be fully updated to that change. Maybe I can do that in the context of *this* project. First some more reorientation.

'Program Software System' now looks 'outdated', compared to the programming work described in 'Document Software System'.

I might actually move many of those topics from 'Future\Interesting Now' to 'Postponed'.

I also would want to put a cut into all the project docs and all the circle docs: this is the language and this is the rest.

So it gets isolated. In the past I wanted to put everything I did (and will ever do) with software development at home in a single system so general that I called it 'Software System'. Many docs are general and describe both that language + OS-like topics and applications. I might want to cut that in two: language and the rest. I might like to open source the language at one point and just leave the rest out of it.

I think I interwove these things maybe a little too much. I just liked to subdivide things into a single system of subdivision into which everything fitted. Also, the interweaving may have been stimulated by my wanting to combine this 'Creator' project with the 'Circle' language project. The 'Creator' project was about model-driven development, aspects and framework more than being a real computer language. I wanted to combine the two things into a single system, so that may have lead me to try and put everything into a single system. Now, I think I know that Circle is the computer language and you could program model-driven aspect oriented software with it, if it can provide the aspect construct. Really, I think it helps to not try and solve all problems at once.

### 2019-08-04 Done Notes converting version folders to source control history

This is a lot of work.

* Exp: Search for XXXX in the folder "Circle Docs\1. Language".
* Obs: 54 matches.
* Hyp: 54x a version folder structure to convert to source control history.
* Less than I thought and probably doable.
* Maybe rename all commits, putting the version number in front, so it is more apparent that it is alternative source control history.