Automating font production using DTL FontMaster

- *Frank E. Blokland*

Font production process:

who is involved? (i.e. the market)

Font production process: who is involved? (i.e. the market)

- Type designers

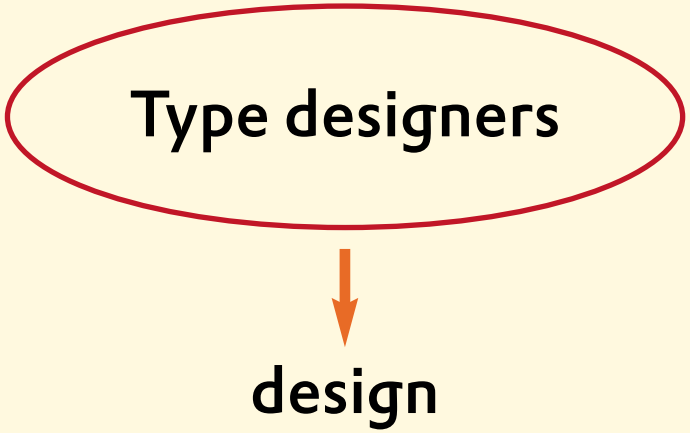
Font production process: who is involved? (i.e. the market)

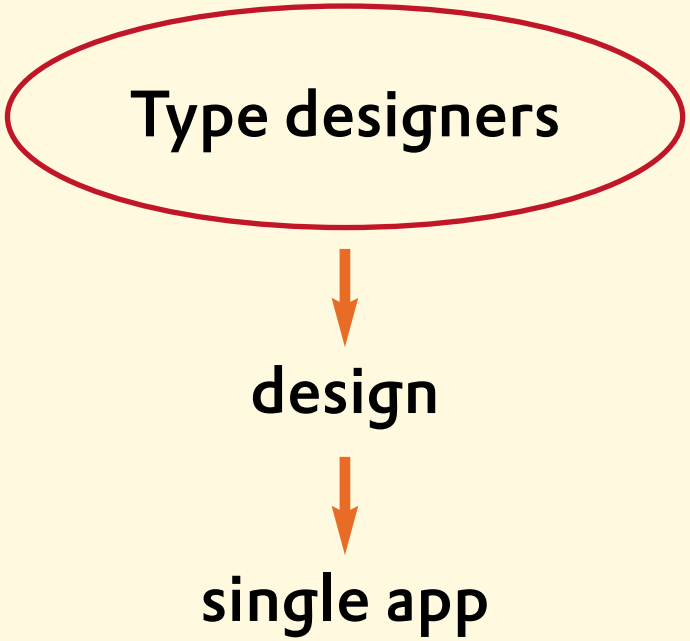
- Type designers
- Font producers

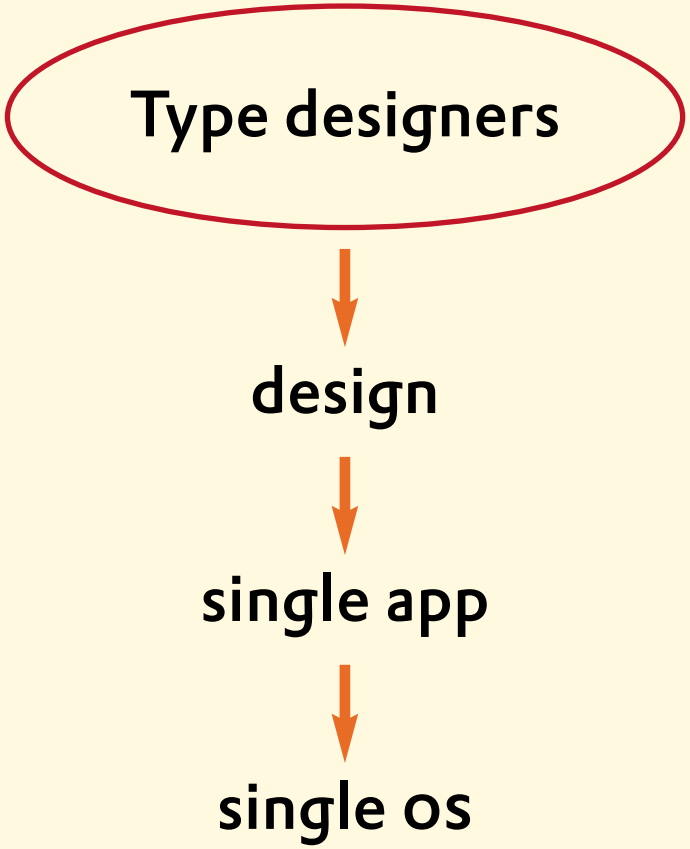
Font production process: who is involved? (i.e. the market)

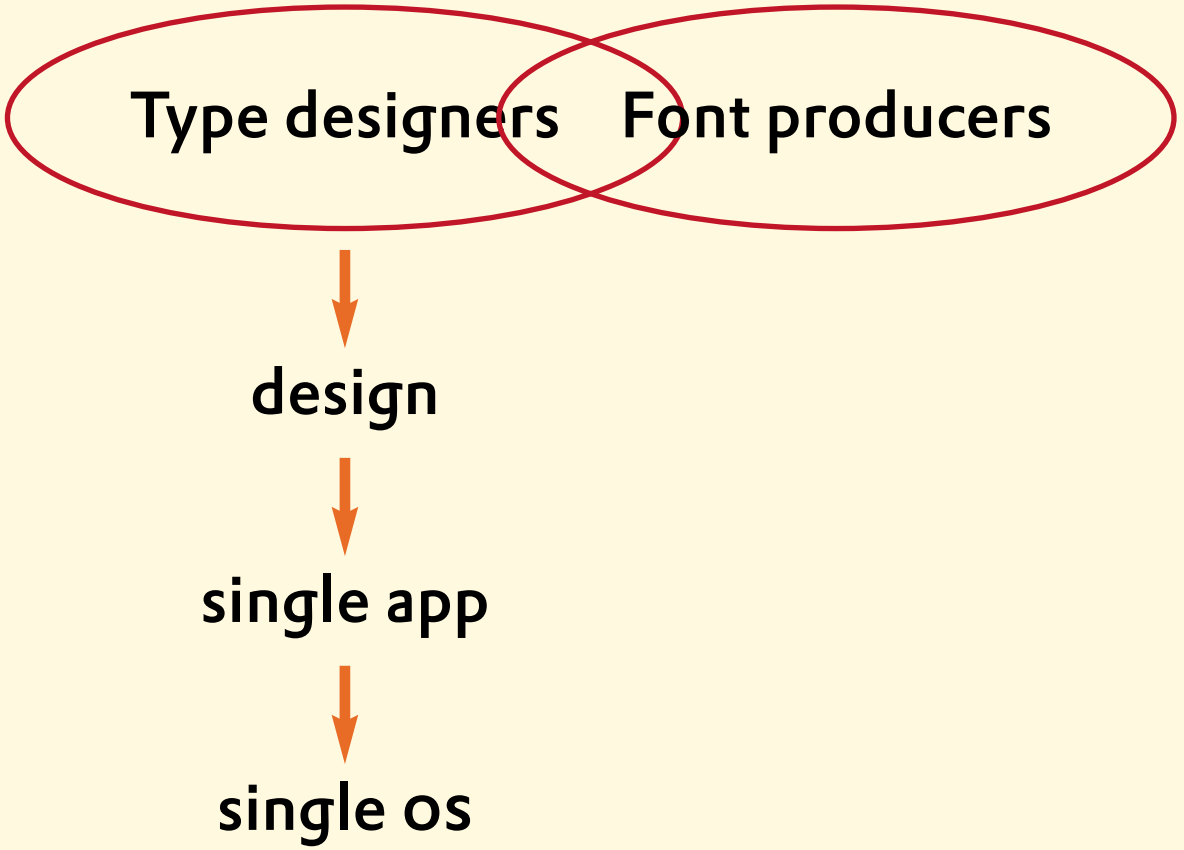
- Type designers
- Font producers
- Programmers

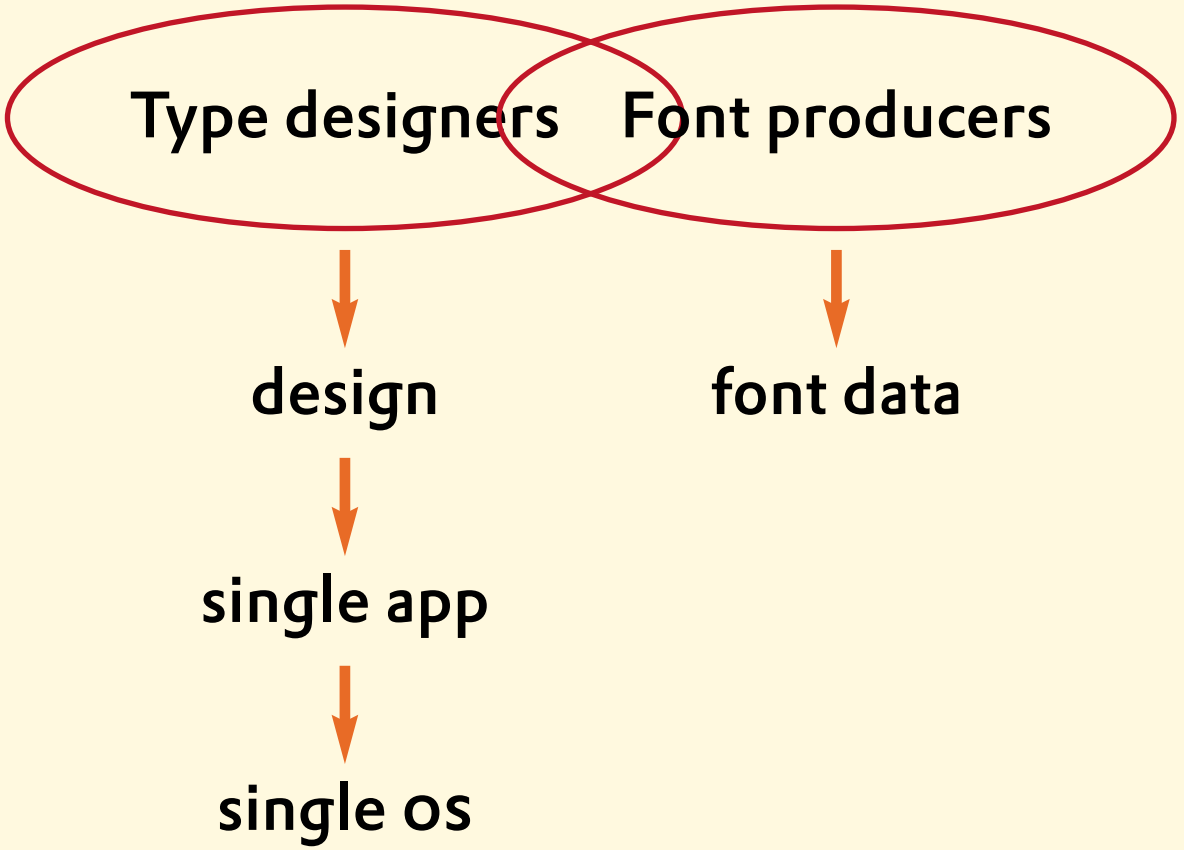
Type designers

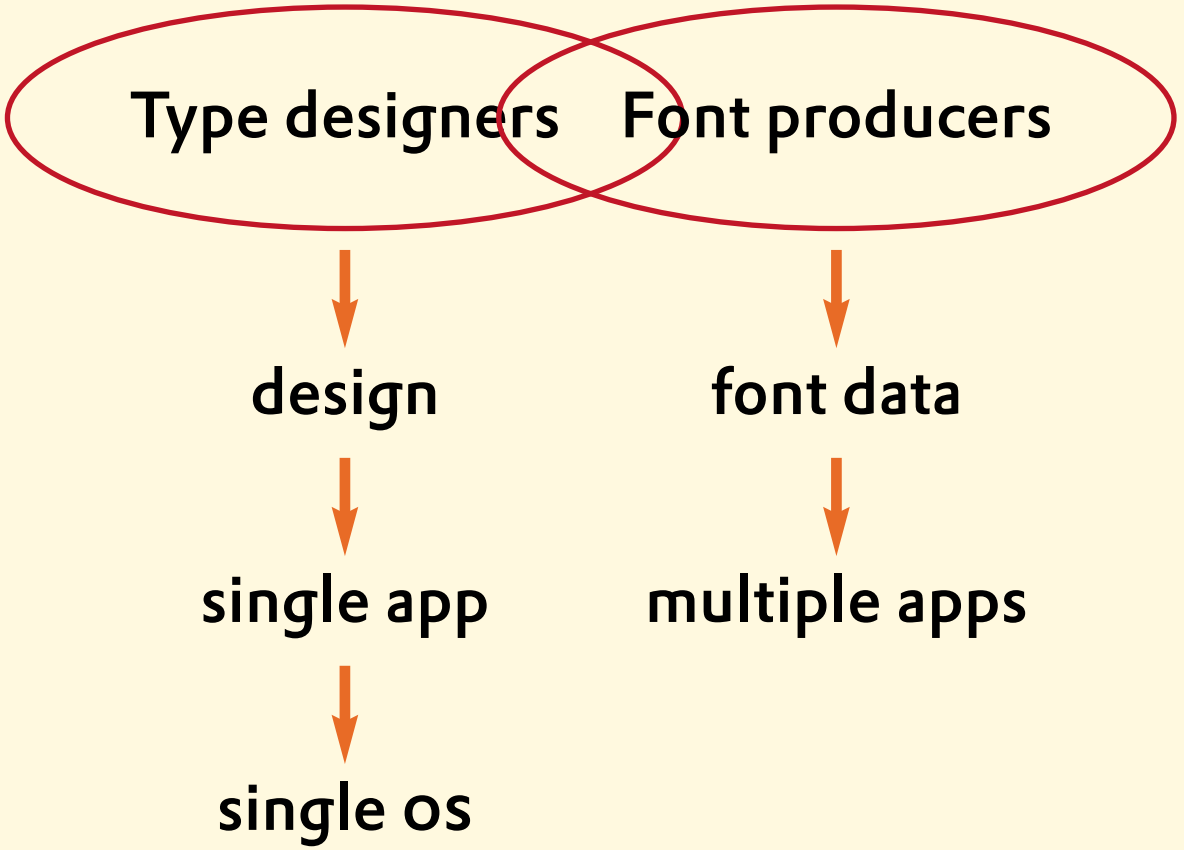

Type designers
design

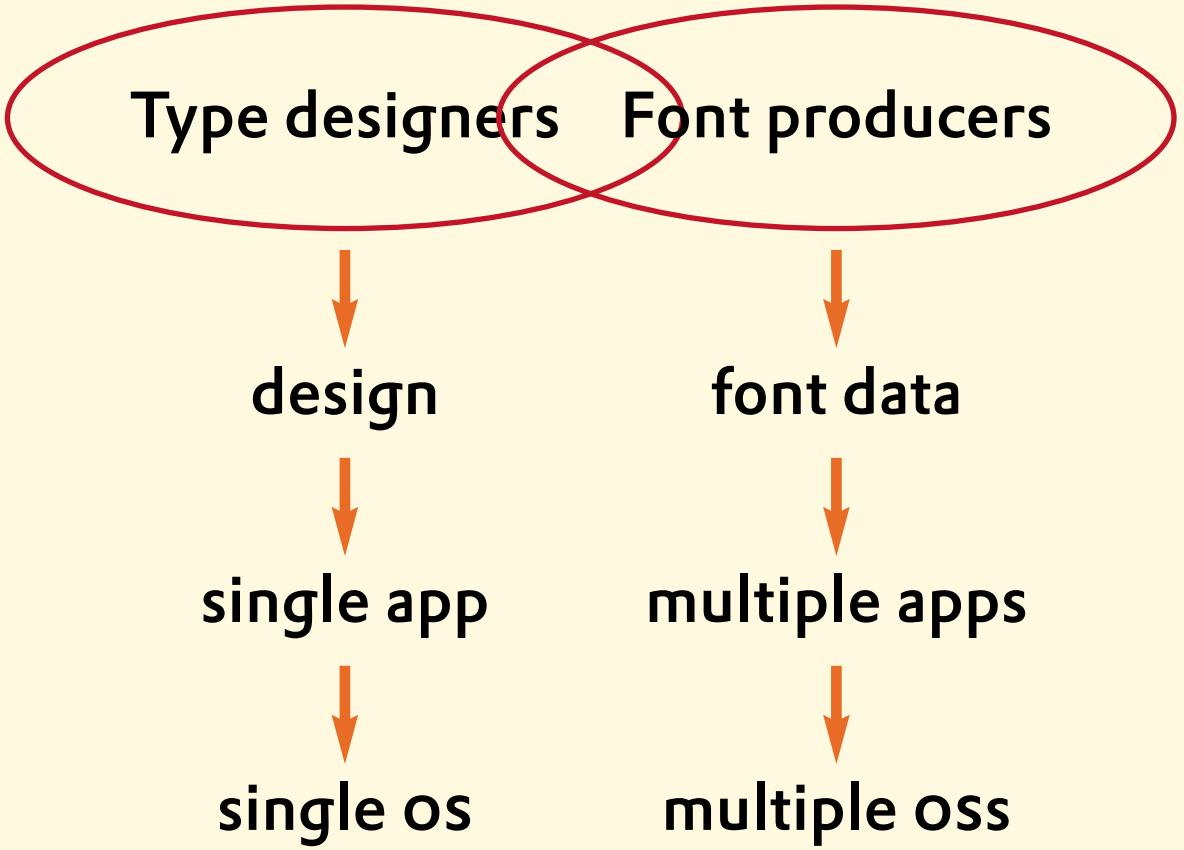

Type designers
design
single app

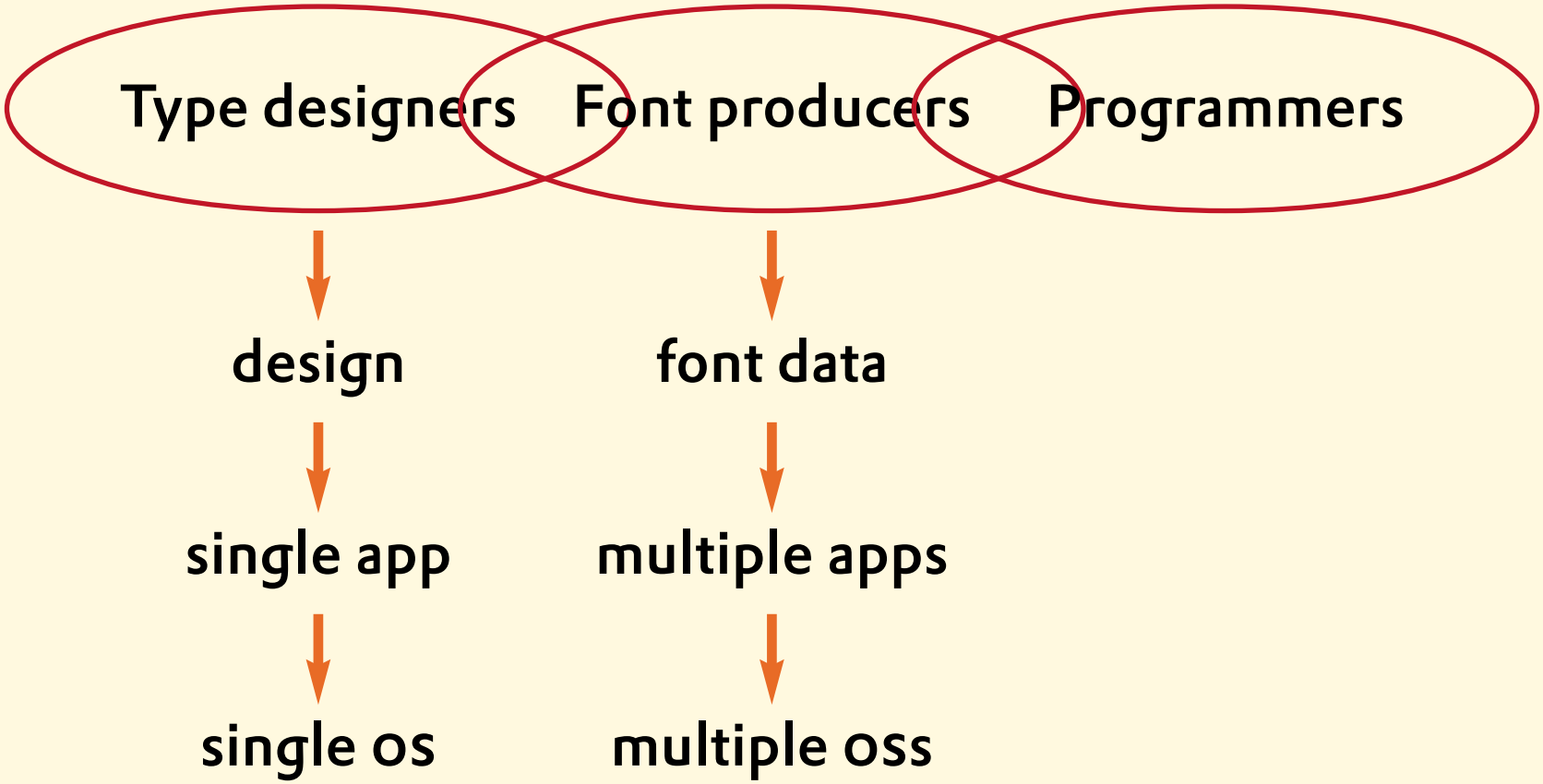

Type designers
design
single app
single os

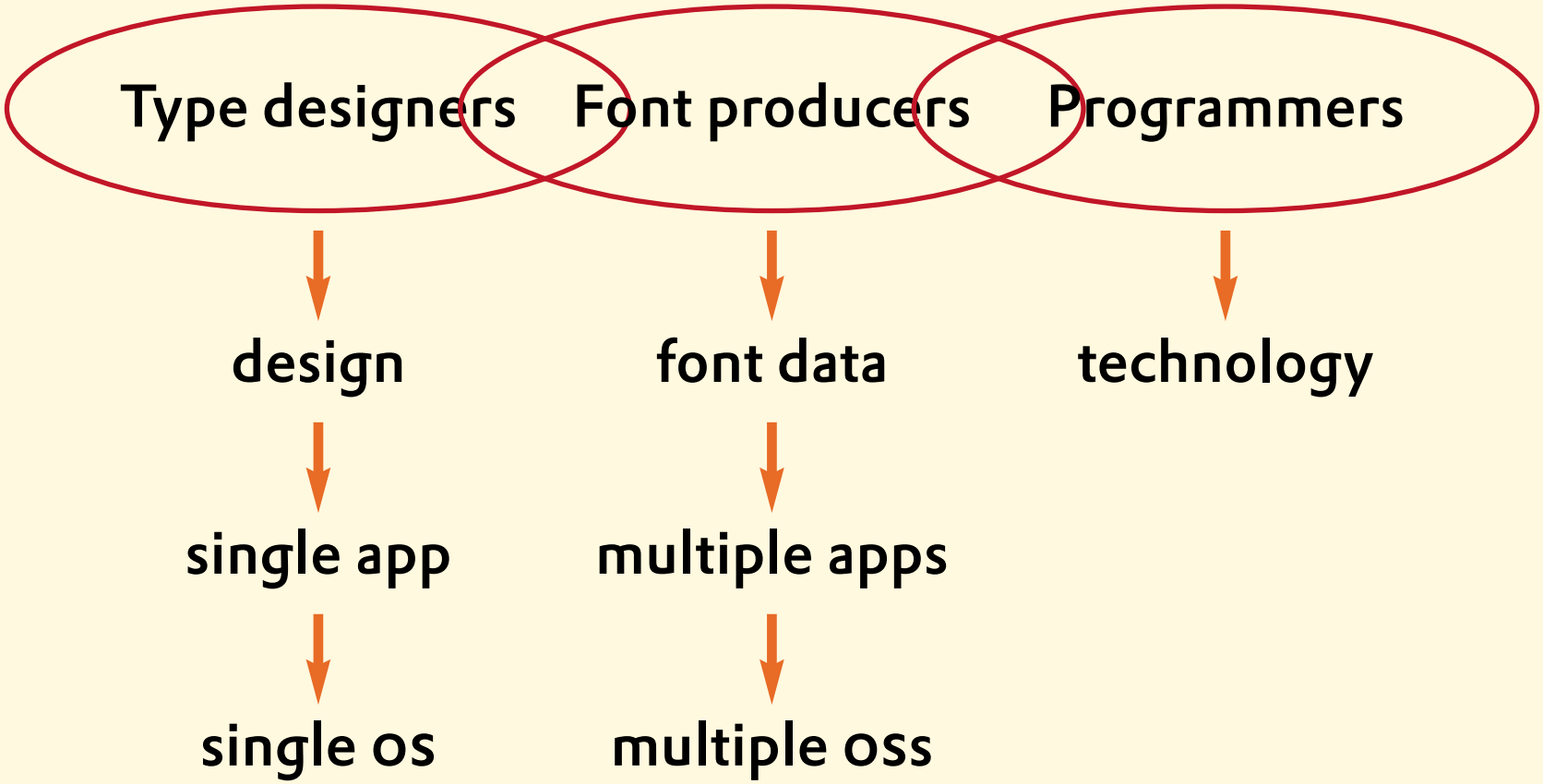

Type designers
Font producers
design
single app
single os

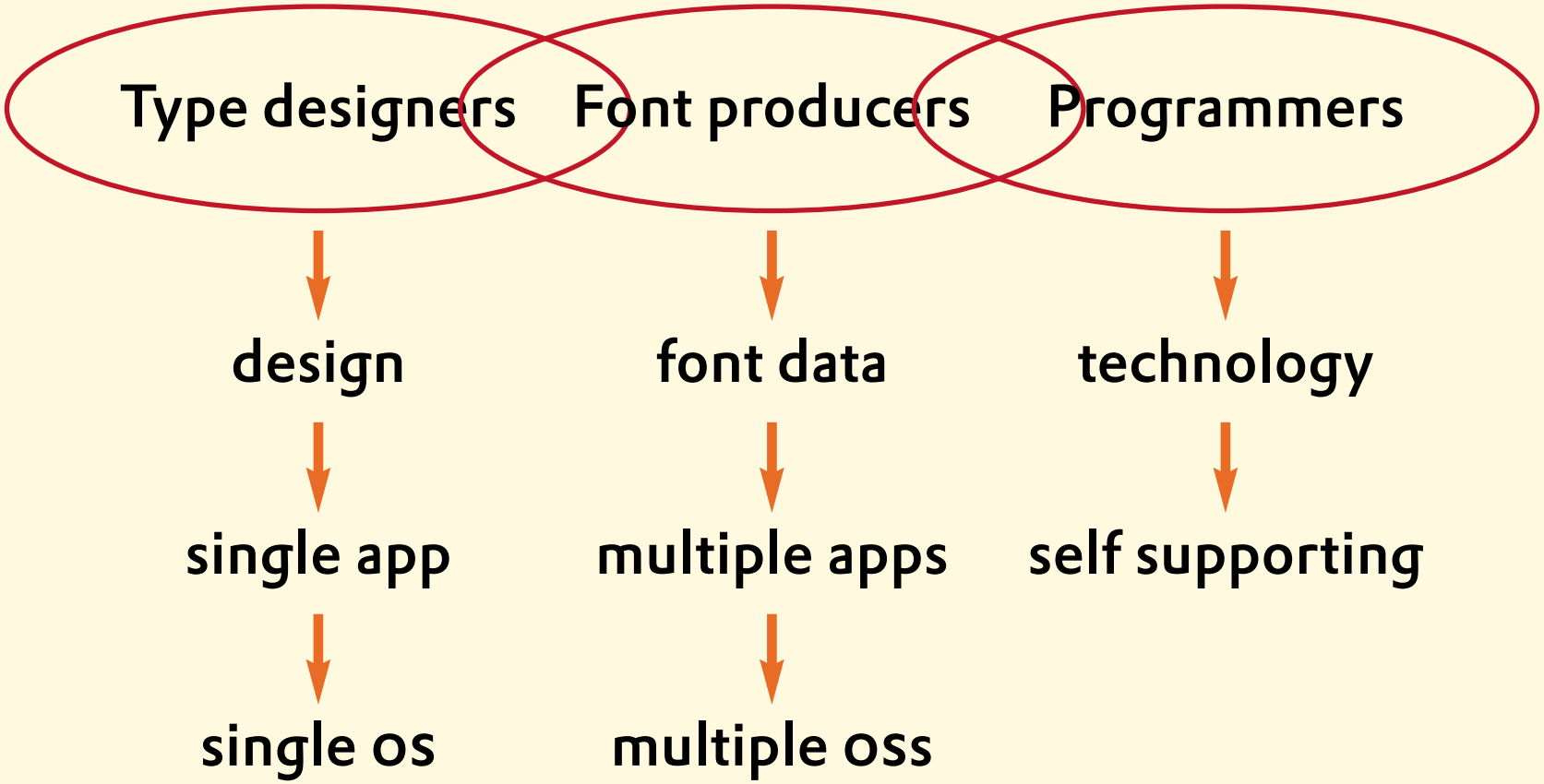

Type designers
Font producers
design
font data
single app
single os

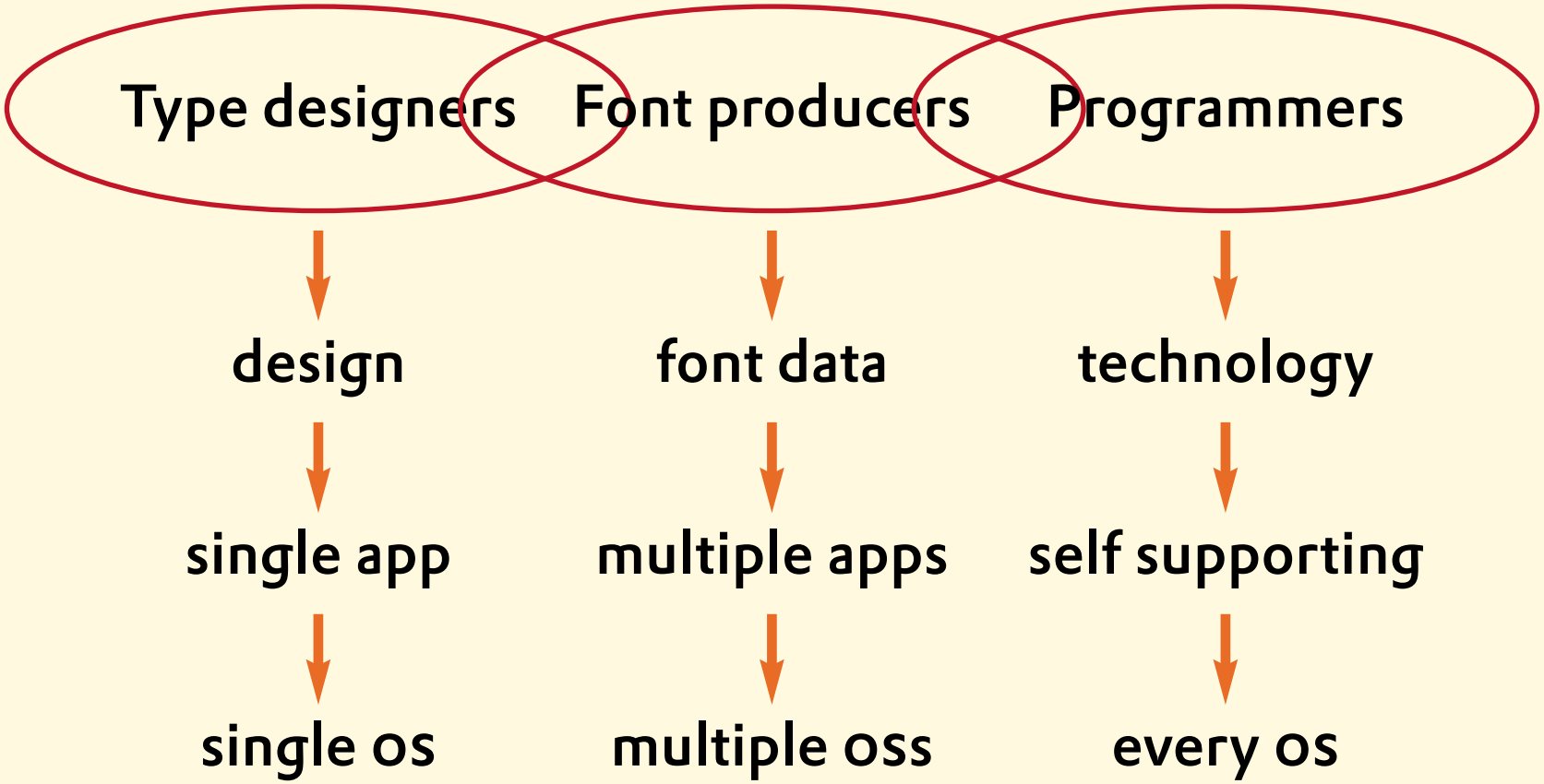

Type designers
Font producers
design
font data
single app
multiple apps
single os

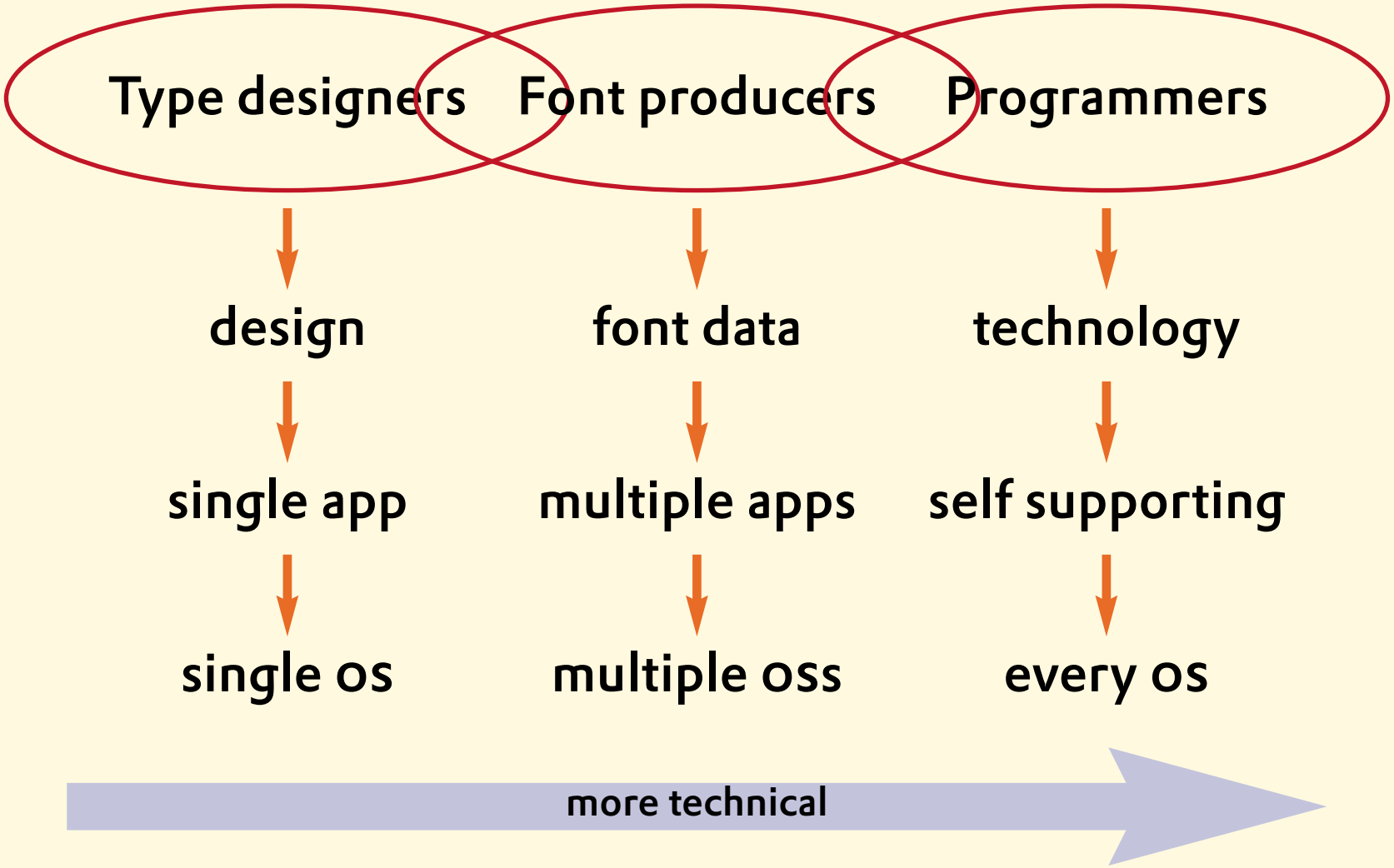

Type designers
Font producers
design
font data
single app
multiple apps
single os
multiple oss


Type designers
Font producers
Programmers
design
font data
single app
multiple apps
single os
multiple oss


Type designers
Font producers
Programmers
design
font data
technology
single app
multiple apps
single os
multiple oss


Type designers
Font producers
Programmers
design
font data
technology
single app
multiple apps
self supporting
single os
multiple oss


Type designers
Font producers
Programmers
design
font data
technology
single app
multiple apps
self supporting
single os
multiple oss
every os


Type designers
Font producers
Programmers
design
font data
technology
single app
multiple apps
self supporting
single os
multiple oss
every os
more technical

Font production process:

how to automate?

Font production process: how to automate?

- Support for scripting (Perl, Python)

Font production process:

how to automate?

- ~~Support for scripting (Perl, Python)~~
- Extensive build in batch functionality supported by simple command language

Font production process:

what to automate?

Font production process: what to automate?

- Design stage

Font production process: what to automate?

- Design stage
- Editing stage

Font production process: what to automate?

- Design stage
- Editing stage
- Data management stage

Font production process: what to automate?

- Design stage
- Editing stage
- Data management stage
- Font generation stage

Font production process: what to automate?

- Design stage
- Editing stage
- Data management stage
- Font generation stage
- Format enhancement stage

1. Design stage

1. Design stage

- Sketching/drawing on paper, subsequently manual digitizing (*IkarusMaster*) and/or auto tracing (*TraceMaster*)

1. Design stage

- Sketching/drawing on paper, subsequently manual digitizing (*IkarusMaster*) and/or auto tracing (*TraceMaster*)
- Sketching/drawing on screen, directly in a glyph editor (*BezierMaster*/*IkarusMaster*) or using a program like Adobe Illustrator®

2. Editing stage

2. Editing stage

- Enhancement (æsthetically and technically) of the design (*BezierMaster*/*IkarusMaster*)

2. Editing stage

- Enhancement (æsthetically and technically) of the design (*BezierMaster*/*IkarusMaster*)
- Enhancement of the glyph set, from basic character set to multiple code pages support (*BezierMaster*/*IkarusMaster*)

2. Editing stage

- Enhancement (æsthetically and technically) of the design (*BezierMaster/IkarusMaster*)
- Enhancement of the glyph set, from basic character set to multiple code pages support (*BezierMaster/IkarusMaster*)
- Spacing (*BezierMaster/IkarusMaster*)

2. Editing stage

- Enhancement (æsthetically and technically) of the design (*BezierMaster*/*IkarusMaster*)
- Enhancement of the glyph set, from basic character set to multiple code pages support (*BezierMaster*/*IkarusMaster*)
- Spacing (*BezierMaster*/*IkarusMaster*)
- Kerning (*KernMaster*/*Bezier-* and *IK Master*)

3. Data management stage

3. Data management stage

- Database building/enhancing/merging (*BezierMaster*/*IkarusMaster*)

3. Data management stage

- Database building/enhancing/merging (*BezierMaster*/*IkarusMaster*)
- Consistency checking (*ContourMaster*)

3. Data management stage

- Database building/enhancing/merging (*BezierMaster*/*IkarusMaster*)
- Consistency checking (*ContourMaster*)
- Technical quality control (*ContourMaster*)

4. Font generation stage

4. Font generation stage

- Producing different font formats for different platforms (*DataMaster*)

4. Font generation stage

- Producing different font formats for different platforms (*DataMaster*)

Numbers 1–4: Long term technology; resulting in a glyph set wherefrom different font formats can be generated

5. Format enhancement stage

5. Format enhancement stage

- Delta hinting (VTT)

5. Format enhancement stage

- Delta hinting (VTT)
- Adding OT features (*DataMaster*/VOLT)

5. Format enhancement stage

- Delta hinting (VTT)
- Adding OT features (*DataMaster*/VOLT)

Number 5: Short term technology; meant to adapt font data to actual reproduction technology

Data structure of FM

A versatile system for batch production:

Data structure of FM

A versatile system for batch production:

- Glyph database (platform independent)

Data structure of FM

A versatile system for batch production:

- Glyph database (platform independent)
- UFM file (font naming/metrics)

Data structure of FM

A versatile system for batch production:

- Glyph database (platform independent)
- UFM file (font naming/metrics)
- AFM/kern.fea file (kerning information)

Data structure of FM

A versatile system for batch production:

- Glyph database (platform independent)
- UFM file (font naming/metrics)
- AFM/kern.fea file (kerning information)

- Character layout file

Data structure of FM

A versatile system for batch production:

- Glyph database (platform independent)
- UFM file (font naming/metrics)
- AFM/kern.fea file (kerning information)

- Character layout file
- OpenType layout features file

Data structure of FM

Conditions for OT fonts with layout features:

- The glyphs must be in the database
- The characters must be listed in the character layout file (.cha)
- The characters must be identically named in the features and character layout files

Data structure of FM

Batch functionality via:

Data structure of FM

Batch functionality via:

- Direct parameter input

Data structure of FM

Batch functionality via:

- Direct parameter input
- Command file

Font production process: what to automate?

- Design stage
- Editing stage
- Data management stage
- Font generation stage
- Format enhancement stage

1. Design stage

- Auto tracing /batch (*TraceMaster*)

1. Design stage

- Auto tracing /batch (*TraceMaster*)
- Consistency checking /batch single font (*Bezier*- and *IkarusMaster*)

1. Design stage

- Auto tracing /batch (*TraceMaster*)
- Consistency checking /batch single font (*Bezier*- and *IkarusMaster*)
- Outline quality control /batch single font (*Bezier*- and *IkarusMaster*)

2. Editing stage

- Data management /batch:
  merge composites (text file),
  metrics handling (text file)
  (*Bezier*- and *IkarusMaster*)

2. Editing stage

- Data management /batch:
  merge composites (text file),
  metrics handling (text file)
  (*Bezier*- and *IkarusMaster*)
- Auto kerning /batch multiple fonts:
  output AFM file & kern feature file
  (*KernMaster*)

3. Data management stage

- Data management /batch:
  merge multiple databases /command file
  (*Bezier*- and *IkarusMaster*)

3. Data management stage

- Data management /batch:
  merge multiple databases /command file
  (*Bezier*- and *IkarusMaster*)
- Consistency checking /batch multiple fonts
  (*ContourMaster*)

3. Data management stage

- Data management /batch:
  merge multiple databases /command file
  (*Bezier*- and *IkarusMaster*)
- Consistency checking /batch multiple fonts
  (*ContourMaster*)
- Outline quality control /batch multiple fonts
  (*ContourMaster*)

3. Data management stage

- Interpolation /batch multiple fonts (*BlendMaster*)

4. Font generation stage

- Font generation /single fonts:

  Automatic OT layout features generation

  (*Bezier*- and *IkarusMaster*)

4. Font generation stage

- Font generation /single fonts:
  Automatic OT layout features generation
  (*Bezier*- and *IkarusMaster*)
- Font generation /batch multiple fonts:
  Automatic OT layout features generation
  Output optionally via Command file(s)
  (*DataMaster*)

Demonstrations

- FM Track, Thursday 28 September

Automating font production using DTL FontMaster

- *Frank E. Blokland*