File Repair (CPP-027)

CPP-Identifier	CPP-027
CPP-Label	File Repair
Author	Bertrand Caron
Contributors	Johan Kylander
Evaluators	Matthew Addis, Felix Burger, Maria Benauer
Date of edition completed	29.08.2025
Change history	**Comments**
Version 1.0 - 29.08.2025	Milestone version

1. Description of the CPP

As a way of handling validation errors or rendering errors, the TDA corrects structural issues identified in its *Files*.

Inputs and outputs

Input(s)	
Data	*File*
	sometimes: *Representation*, whose format structures are causing issues during validation, rendering or use
Metadata	Properties of source *File* or *Representation*
Documentation / guidance	*Preservation action registry*
	Format specifications
Output(s)	
Data	*File* or *Representation* whose format structures are deemed "fixed" *and/or* *File* or *Representation* whose erroneous format structures are considered acceptable after further investigation
Metadata	Properties of target *File* or *Representation*
	New/updated *File* or *Representation* properties
	Provenance metadata
	Optional: file comparison between source and target *File* / *Representation*.
Documentation / guidance	Optional: updated *Preservation action registry*

Definition and scope

File Repair is an operation performed to correct erroneous format structures that could prevent the expected use or rendering of the *File* by the repository or the user[1]. In addition,

[1] For additional considerations on the topic, see this conference paper:Lindlar, Micky. "NOT WELL-FORMED OR INVALID. NOW WHAT?: Towards a Formalized Workflow for Format Validation Error Treatment." *IPRES 2023*, 2023. https://hdl.handle.net/2142/121092.

erroneous format structures can potentially hinder preservation actions, such as file format migration, as tools developed to process the *Files* expect the *Files* to have a defined structure.

Erroneous format structures have many different causes. Some examples are listed below:

- Improper behaviours of creation software (e.g. InDesign, in some previous versions, was known to export corrupted PDFs);
- Partial or failed transfers (e.g. digital photographs transferred through an unstable bluetooth connection);
- Corruption or loss when data is transferred to a TDA or is processed within a TDA where there are no 'good' copies of the data that can be recovered from and hence file repair is necessary (e.g. a *File* is corrupted during transfer to a TDA which goes undetected or a *File* is corrupted during ingest and is then replicated across all copies held by the TDA).
- Edition with a different software than the creation tool (e.g. PDFs produced with pdfTeX, then edited with Quark XPress);
- etc.

The process is triggered either by a) a failed attempt to render or use the *File* with a rendering software (or any software that takes the *File* as input and makes use of it in any way), b) warnings / errors returned by a validation tool, or c) fixity checks revealing that all copies of a *File* in a TDA have been damaged in some way.

The impact of the erroneous format structures should be assessed as "low", "medium" or "high" according to the following criticality / severity criteria:

- Impact is considered "High" when important parts of the *File's* or *Representation's* content are not accessible;
- Impact is considered "Medium" when a regular feature of the *File* or *Representation* is not supported, either for the consumer or the TDA (e.g. if metadata extraction cannot be performed);
- Impact is considered "Low" when no effect can be detected on the *File's* or *Representation's* content or features.

A *File* or *Representation* might be affected by several deviations from the standard. In that case, the impact should be assessed for each identified deviation. The process should then try to address issues from the most to the least impactful. One should keep in mind that errors might be related, so that one repairing operations might affect several errors returned by a validation tool. In addition, some tools would stop the validation process at the first error, so that repairing the initial error returned by the tool would lead to a still invalid *File* for another reason.

This CPP covers the following steps:

- The initial analysis step that first determines precisely which erroneous format structure is responsible for the failure using validation tools and metadata extractors, and then evaluates the risk that these structures pose for current and future users;
- The intervention on the *File* itself;
- The control step that measures the impact of repair, and controls that no significant properties were affected in the process;
- The documentation step that records the details of the process;
- The handling of false positives and identification of gaps in the tools producing the false positive warnings / errors.

Many reasons could cause the process not to be effectively performed. In the description below, three are mentioned:

- The impact of erroneous structures is deemed low compared to the *Object* value and the operation's complexity;

- The deviation from the standard could not be assessed;
- A repair method could not be provided by CPP-012 (**Risk Mitigation**).

In any of these cases, the file repair process would end and another error-handling option among the ones suggested in CPP-010 (**File Format Validation**) should be considered.

File repair is a manual and investigative process that requires in-depth knowledge of format structures. Therefore, its performance requires a high level of institutional maturity and organisational resources in digital preservation and is recommended only for mature TDAs.

Process description

Trigger event(s)

Trigger event	CPP-identifier
Error or warning returned by the validation process	CPP-010 (File Format Validation)
Failed attempt to render or use in any way	/
Corrupted *File* or *Representation*, no intact copy available	CPP-004 (Data Corruption Management)

Step-by-step description

No	Supplier	Input	Steps	Output	Customer
1		Observation of failure to render or use a *File* or *Representation*	Use rendering tools and edition tools to check whether the erroneous format structures impact the *File's* content and features	Effect on the *File's* content and features	
		Source *File* or *Representation*			
		Any software that takes the format as input, in particular rendering tools			
2		Error message	If Format validation has returned that the *File* is invalid, check whether knowledge bases on	Impact as assessed by external resources	

		Literature or knowledge bases on validation errors[2]	validation errors already have assessed the erroneous format structures' impact		
3		Knowledge gathered in the previous step(s)	Evaluate current impact of erroneous format structures (low, middle, high…) against significant properties[3]. If the impact is evaluated as "low", consider not proceeding further.	Risk impact	
	CPP-022 (Significant Properties Definition)	Significant properties			
4		Data: source *File* / *Representation*	Use different validation tools and metadata extractors to identify precisely which format structures are responsible for the detected issue	Errors and warnings	
	CPP-010 (File Format Validation)	Tools: as many validation tools as possible; as a default, reference parsing tools associated with the format		Source *File* or *Representation* properties	
	CPP-009 (Metadata Extraction)	Tools: metadata extractors			
5		Errors and warnings	Compare with format specifications to determine the distance between erroneous *File*	Estimated deviation from the specification	

[2] An example of such a knowledge base is the JHOVE wiki that associates an impact assessment and remediation suggestion.
[3] For example, if the erroneous format structure affects the Image Unique Identifier (EXIF tag) and that the presence of internal technical metadata is not deemed significant by the TDA, the impact will be deemed low.

		Format specifications	structures and specification requirements		
6			If there is no deviation, consider contributing to improvement of software (rendering, validation, etc.) that returned the false positive, and do not proceed further. If the deviation from the standard cannot be assessed, consider not proceeding further.	Issues / contributions to tools	
7		Errors and warnings Source *File* or *Representation* properties Estimated deviation from the specification	Revise results of step 1 in light of outputs of steps 2 and 3, then go to step 6	Revised risk impact	
8		Revised impact level and consequence	Decide whether or not to perform file repair		
9	CPP-012 (Risk Mitigation)	*Preservation action registry*	Request a repair method, if no epair method is available, consider not proceeding further	Repair method	
10		Tool: edition tool / hexadecimal editor	Perform chosen repair method	Target *File* or *Representation*	

11	CPP-009 (Metadata Extraction), CPP-010 (File Format Validation)	Data: Target *File* or *Representation* Tools: metadata extractors and validation tools used in step 2	Run metadata extraction and file format validation on the target *File* or Representation.	Target *File* or *Representation* properties	
12	CPP-022 (Significant Properties)	Target *File* or *Representation* properties (in particular validity status) *Significant properties*	Control that the target *File* or *Representation* has expected validity status and significant properties If the *File* or *Representation* is still invalid, assess the outcome of the repair method: - If the repair method was successful (i.e., fixed indeed an erroneous format structure), resume the process from step 1 with the target *File* or *Representation*; - If the repair method was unsuccessful, resume the process from step 7 with the source *File* or *Representation*	Decision whether the repair method should be confirmed	
13			Decide which *File* or *Representation* should be retained (source, target or both) If target *File* or *Representation* is to be retained, proceed with		

			further steps. If not, the process ends		
14			Document precisely the nature of the operation performed (tool and version used, date, options or command-line arguments, etc.)	New *Provenance metadata*	
15			Optional: Update the Preservation action registry with the new process description	New or updated repair method	CPP-012 (Risk Mitigation)

Rationale(s)[4] and worst case(s)

Rationale	Impact of inaction or failure of the process
Erroneous format structures can cause rendering issues or prevent expected use of the *File* or *Representation*. In order to decide whether to perform the repair method, TDA should weigh the impact and likelihood of the risk against the Information *Object* value, the operation complexity and available knowledge of its staff.	Potentially, inability to render the *File*. See Caron, Bertrand. "'Validation OK' They Said: Fixing the Rendering of a So-Called Valid PDF." *Open Preservation Foundation* (blog), May 21, 2025. https://openpreservation.org/blogs/validation-ok-they-said-fixing-the-rendering-of-a-so-called-valid-pdf/.
If the TDA is performing format validation, having defined procedures to handle errors by investigating their cause and correcting structures is necessary as a proper response.	Without comparative support, the know-how or the resources required to manage the investigation process and repair actions can get lost, resulting in *Files* either remaining invalid and at a risk of being preserved properly, or not even ingested at all.
It is recommended that a TDA maintains an institutional Preservation Actions Registry that includes common validation errors and potential solutions to fix them.	

2. Dependencies and relationships with other CPPs

Dependencies

CPP-ID	CPP-Title	Relationship description
CPP-010	File Format validation	Validation is generally implemented as part of an ingest workflow. If validation fails, a decision can be taken to ingest the *File* or *Representation* as is, reject it or repair the erroneous structures.
CPP-012	Risk Mitigation	Risk Mitigation, as the provider of all actions aiming at limiting the impact or likelihood of identified risks, is intended to prescribe appropriate methods to repair the *File* or *Representation*.

[4] Term derived from PREMIS.

CPP-022	Significant Properties Definition	Modifying the *File's Bitstream* in order to correct faulty structures can cause unexpected changes to the rendering and behaviour of an *Object*. Thus file repair should also be evaluated based on significant properties.
CPP-005	Identifier Management	Soft dependency (i.e. may require): A repaired *File* may get a new identifier.
CPP-009	Metadata Extraction	Soft dependency (i.e. may require): Tools extracting properties of the *File* or *Representation* are useful, if not necessary, for analysing which format structures are erroneous.

Other relations

Relation	CPP-ID	CPP-Title	Relationship description
Triggers	CPP-017	Disposal	The file repair process may end up with a situation where the *File* cannot be repaired. In such cases, disposal is required by the file repair process.
Required by	CPP-013	Object Management Reporting	Fixing invalid *Files* produces *Provenance metadata* and data for quality reporting to the stakeholders
Affinity with	CPP-014	File Format Migration	Both operations affect the *Files' Bitstream* and aim at reducing the risk. Migration is typically performed to mitigate an obsolescence risk, while repair handles format structure issues.
Alternative to	CPP-004	Data Corruption Management	In some cases where no intact copy of corrupted or broken data is available, repairing the structure of an altered copy is the only option.

3. Links to frameworks

Certification

Certification framework	Term used in framework to refer to the CPP	Section
CTS Link	"The approach to resolving issues e.g. whether the digital objects are (...) fixed by the	R10 "Quality assurance"

	repository”	
Nestor Seal Link	/	/
ISO 16363 Link	/	/

Other frameworks and reference documents

Reference Document	Term used in framework to refer to the process	Section
OAIS	/	/
PREMIS	Format validation	Section Format Information, p. 249.

4. Reference implementations

Example use cases

Improperly formatted dates in a PDF

Institutional Background	
Institution	Leibniz Information Centre for Science and Technology, Germany
Hyperlink	Lindlar, Micky. “A Date with PDF-HUL-133 ‘Improperly Formed Date.’” *Open Preservation Foundation* (blog), May 13, 2024. https://openpreservation.org/blogs/a-date-with-pdf-hul-133-improperly-formed-date/.
Description	
Trigger event	A PDF *File* that was to be ingested was analyzed by TIB and errors were returned by validation tools (JHOVE, pdfcpu and qpdf).
Problem statement	The creation and modification dates are incorrectly formatted, and as a consequence rendering tools cannot display these dates in their metadata panel.
Proposed solution	After analysing the formatting issue and comparing it to the expected format from the specification, TIB compared two different repair methods, applied them and compared the result

	between source and target *Files*. The final decision was to retain only the source format, as the risk associated with this format structure issue was deemed low.

Repairing a PDF that causes Rendering Issues in Adobe Reader

Institutional Background	
Institution	TIB – Leibniz Information Centre for Science and Technology and University Library, Germany
Hyperlink	Caron, Bertrand. "'Validation OK' They Said: Fixing the Rendering of a So-Called Valid PDF." *Open Preservation Foundation* (blog), May 21, 2025. https://openpreservation.org/blogs/validation-ok-they-said-fixing-the-rendering-of-a-so-called-valid-pdf/.
Description	
Trigger event	A PDF *File* that was to be ingested was identified by TIB could not be displayed by Adobe Reader.
Problem statement	The analysis used validation tools to identify the erroneous format structure: very high values for integers, for some properties where smaller values were expected by the rendering tools.
Proposed solution	Though no deviation was noticed from the specification, TIB considered the risk as high as Adobe Reader is the default rendering tool for PDFs. After performing the repair, significant properties (textual content, visual aspect) were checked with the tool comparepdf to confirm that no important information or feature was lost in the process. **After noticing that the process caused the loss of unreferenced PDF *Objects*** (undisplayed but potentially meaningful pieces of information located in the PDF *File*) that were considered as a significant property**, the repair method was corrected.** The decision was to retain only the repaired *File*.

Publicly available documentation

Institution	Organisation type	Language	Hyperlink
TIB – Leibniz Information Centre for Science and Technology and University Library, Germany	National library	English	Part of all ingest routines after validation stack process, see: https://wiki.tib.eu/confluence/spaces/lza/pages/93608618/Ingest?preview=/93608618/310285406/mets-deposit.png#Ingest-PAIProcessdiagrammMETSdeposit, https://wiki.tib.eu/confluence/spaces/lza/pages/93608618/Ingest?preview=/93608618/310285409/csv_deposit.png#Ingest-CSVProcessdiagrammCSVdeposit and https://wiki.tib.eu/confluence/spaces/lza/pages/93608618/Ingest?preview=/93608618/310285408/oai_deposit.png#Ingest-OAIDpsProcessdiagrammOAI-PMHdeposit
	Non-commercial digital preservation service		
	Research infrastructure		
	Research performing organisation		